```
IIIIIIIII      NNN          NNN  PPPPPPPPPPPP        SSSSSSSSSSS  MMM         MMM  BBBBBBBBBBB
IIIIIIIII      NNN          NNN  PPPPPPPPPPPP        SSSSSSSSSSS  MMM         MMM  BBBBBBBBBBB
IIIIIIIII      NNN          NNN  PPPPPPPPPPPP        SSSSSSSSSSS  MMM         MMM  BBBBBBBBBBB
   III         NNN          NNN  PPP         PPP  SSS             MMMMMM    MMMMMM  BBB        BBB
   III         NNN          NNN  PPP         PPP  SSS             MMMMMM    MMMMMM  BBB        BBB
   III         NNN          NNN  PPP         PPP  SSS             MMMMMM    MMMMMM  BBB        BBB
   III         NNNNNN       NNN  PPP         PPP  SSS             MMM  MMM  MMM  BBB        BBB
   III         NNNNNN       NNN  PPP         PPP  SSS             MMM  MMM  MMM  BBB        BBB
   III         NNNNNN       NNN  PPP         PPP  SSS             MMM  MMM  MMM  BBB        BBB
   III         NNN  NNN     NNN  PPPPPPPPPPPP        SSSSSSSSS    MMM       MMM  BBBBBBBBBBB
   III         NNN  NNN     NNN  PPPPPPPPPPPP        SSSSSSSSS    MMM       MMM  BBBBBBBBBBB
   III         NNN  NNN     NNN  PPPPPPPPPPPP        SSSSSSSSS    MMM       MMM  BBBBBBBBBBB
   III         NNN       NNNNNN  PPP                        SSS  MMM       MMM  BBB        BBB
   III         NNN       NNNNNN  PPP                        SSS  MMM       MMM  BBB        BBB
   III         NNN       NNNNNN  PPP                        SSS  MMM       MMM  BBB        BBB
   III         NNN          NNN  PPP                        SSS  MMM       MMM  BBB        BBB
   III         NNN          NNN  PPP                        SSS  MMM       MMM  BBB        BBB
   III         NNN          NNN  PPP                        SSS  MMM       MMM  BBB        BBB
IIIIIIIII      NNN          NNN  PPP           SSSSSSSSSSS     MMM       MMM  BBBBBBBBBBB
IIIIIIIII      NNN          NNN  PPP           SSSSSSSSSSS     MMM       MMM  BBBBBBBBBBB
IIIIIIIII      NNN          NNN  PPP           SSSSSSSSSSS     MMM       MMM  BBBBBBBBBBB
```

FILEID**INPSMB

```
IIIIII    NN      NN  PPPPPPPP     SSSSSSSS  MM      MM  BBBBBBBB
IIIIII    NN      NN  PPPPPPPP     SSSSSSSS  MM      MM  BBBBBBBB
  II      NN      NN  PP      PP  SS          MMMM  MMMM  BB      BB
  II      NN      NN  PP      PP  SS          MMMM  MMMM  BB      BB
  II      NNNN    NN  PP      PP  SS          MM  MM  MM  BB      BB
  II      NNNN    NN  PP      PP  SS          MM  MM  MM  BB      BB
  II      NN  NN  NN  PPPPPPPP     SSSSSS    MM      MM  BBBBBBBB
  II      NN  NN  NN  PPPPPPPP     SSSSSS    MM      MM  BBBBBBBB
  II      NN    NNNN  PP                SS  MM      MM  BB      BB
  II      NN    NNNN  PP                SS  MM      MM  BB      BB
  II      NN      NN  PP                SS  MM      MM  BB      BB      ....
  II      NN      NN  PP                SS  MM      MM  BB      BB      ....
IIIIII    NN      NN  PP          SSSSSSSS  MM      MM  BBBBBBBB      ....
IIIIII    NN      NN  PP          SSSSSSSS  MM      MM  BBBBBBBB      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;     1        0001  0 MODULE INPSMB   (%TITLE 'Input symbiont'
;     2        0002  0                  MAIN = INPSMB,
;     3        0003  0                  IDENT = 'V04-000'
;     4        0004  0                  ) =
;     5        0005  1 BEGIN
;     6        0006  1
;     7        0007  1 !
;     8        0008  1 !*****************************************************************************
;     9        0009  1 !*                                                                           *
;    10        0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    11        0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    12        0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;    13        0013  1 !*                                                                           *
;    14        0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
;    15        0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
;    16        0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    17        0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    18        0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    19        0019  1 !*  TRANSFERRED.                                                              *
;    20        0020  1 !*                                                                           *
;    21        0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    22        0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    23        0023  1 !*  CORPORATION.                                                              *
;    24        0024  1 !*                                                                           *
;    25        0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    26        0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27        0027  1 !*                                                                           *
;    28        0028  1 !*                                                                           *
;    29        0029  1 !*****************************************************************************
;    30        0030  1 !
;    31        0031  1
;    32        0032  1 !++
;    33        0033  1 ! FACILITY:
;    34        0034  1 !       Input symbiont.
;    35        0035  1 !
;    36        0036  1 ! ABSTRACT:
;    37        0037  1 !       This is it.
;    38        0038  1 !
;    39        0039  1 ! ENVIRONMENT:
;    40        0040  1 !       VAX/VMS user mode.
;    41        0041  1 !--
;    42        0042  1 !
;    43        0043  1 ! AUTHOR: M. Jack, CREATION DATE: 30-Apr-1982
;    44        0044  1 !
;    45        0045  1 ! MODIFIED BY:
;    46        0046  1 !
;    47        0047  1 !       V03-003 MLJ0115         Martin L. Jack, 29-Jul-1983  13:14
;    48        0048  1 !               Update for $SNDJBC file interface change.
;    49        0049  1 !
;    50        0050  1 !       V03-002 MLJ0113         Martin L. Jack, 26-May-1983  10:21
;    51        0051  1 !               Complete implementation.
;    52        0052  1 !
;    53        0053  1 !       V03-001 MLJ0112         Martin L. Jack, 29-Apr-1983  0:02
;    54        0054  1 !               Track SUBMIT enhancements and SJC name changes.
;    55        0055  1 !
;    56        0056  1 !**
```

```
;    58          0057  1 LIBRARY 'SYS$LIBRARY:LIB';
;    59          0058  1 LIBRARY 'SYS$LIBRARY:TPAMAC';
;    60          0059  1 REQUIRE 'SHRLIB$:JBCPRSDEF';
;    61          0169  1
;    62          0170  1
;    63          0171  1 LITERAL
;    64          0172  1         TRUE=           1,
;    65          0173  1         FALSE=          0;
;    66          0174  1
;    67          0175  1
;    68          0176  1 STRUCTURE
;    69          0177  1         BBLOCK[O,P,S,E;N]=
;    70          0178  1             [N]
;    71          0179  1             (BBLOCK + O)<P,S,E>;
;    72          0180  1
;    73          0181  1
;    74          0182  1 PSECT
;    75          0183  1         CODE=           CODE,
;    76          0184  1         PLIT=           CODE,
;    77          0185  1         OWN=            DATA,
;    78          0186  1         GLOBAL=         DATA;
;    79          0187  1
;    80          0188  1
;    81          0189  1 FORWARD ROUTINE
;    82          0190  1         INPSMB,
;    83          0191  1         PROCESSING_LOOP_HANDLER,
;    84          0192  1         PROCESSING_LOOP,
;    85          0193  1         GET_RECORD,
;    86          0194  1         IDENTIFY_COMMAND_VERB,
;    87          0195  1         GET_LINE_CONTINUATION,
;    88          0196  1         TIMER_AST:              NOVALUE,
;    89          0197  1         FILE_ERROR:             NOVALUE,
;    90          0198  1         MAIN_HANDLER_ACTION,
;    91          0199  1         MAIN_HANDLER;
;    92          0200  1
;    93          0201  1
;    94          0202  1 EXTERNAL ROUTINE
;    95          0203  1         CLI$DCL_PARSE:  ADDRESSING_MODE(GENERAL),
;    96          0204  1         CLI$GET_VALUE:  ADDRESSING_MODE(GENERAL),
;    97          0205  1         CLI$PRESENT:    ADDRESSING_MODE(GENERAL),
;    98          0206  1         LGI$VALIDATE:   ADDRESSING_MODE(GENERAL),
;    99          0207  1         LIB$SFREE1_DD:  ADDRESSING_MODE(GENERAL),
;   100          0208  1         LIB$SIGNAL:     ADDRESSING_MODE(GENERAL),
;   101          0209  1         LIB$TPARSE:     ADDRESSING_MODE(GENERAL);
;   102          0210  1
;   103          0211  1
;   104          0212  1 EXTERNAL
;   105          0213  1         LIB$AB_UPCASE:  ADDRESSING_MODE(GENERAL),
;   106          0214  1         INPSMBCLD;              ! Command tables
;   107          0215  1
;   108          0216  1
;   109          0217  1 EXTERNAL LITERAL
;   110          0218  1         INPSMB$_FACILITY,
;   111          0219  1         INPSMB$_ENTFIL,
;   112          0220  1         INPSMB$_INVCONT,
;   113          0221  1         INPSMB$_INVLOGFIL,
;   114          0222  1         INPSMB$_INVPASS,
```

```
;  115     0223  1          INPSMB$_INVUSER,
;  116     0224  1          INPSMB$_JOBCARD,
;  117     0225  1          INPSMB$_MISSPASS,
;  118     0226  1          INPSMB$_OPENUAF,
;  119     0227  1          INPSMB$_USERVAL;
;  120     0228  1
;  121     0229  1
;  122     0230  1 OWN
;  123     0231  1          CARD_CHANNEL:    WORD,                          ! Channel to card reader
;  124     0232  1          INPUT_FAB:       $FAB_DECL,                     ! FAB for input
;  125     0233  1          INPUT_RAB:       $RAB_DECL,                     ! RAB for input
;  126     0234  1          INPUT_NAM:       $NAM_DECL,                     ! NAM block for input
;  127     0235  1          INPUT_RSA:       VECTOR[NAM$C_MAXRSS,BYTE], ! Resultant string for input
;  128     0236  1          INPUT_UBF:       VECTOR[160,BYTE],              ! Record buffers
;  129     0237  1          OUTPUT_FAB:      $FAB_DECL,                     ! FAB for output
;  130     0238  1          OUTPUT_RAB:      $RAB_DECL,                     ! RAB for output
;  131     0239  1          OUTPUT_NAM:      $NAM_DECL,                     ! NAM block for output
;  132     0240  1          OUTPUT_XAB:      $XABPRO_DECL,                  ! Protection XAB for output
;  133     0241  1          OUTPUT_RSA:      VECTOR[NAM$C_MAXRSS,BYTE], ! Resultant string for output
;  134     0242  1          JOB_LENGTH,                                     ! Length of JOB command
;  135     0243  1          JOB_BUFFER:      VECTOR[80,BYTE],               ! JOB command buffer
;  136     0244  1          PUTMSG_ACTION_ROUTINE,                          ! Action routine for OPCOM or 0
;  137     0245  1          FLAGS:           BBLOCK[4],                     ! General flags
;  138     0246  1          INPUT_COMPLETIONS,                              ! Cards since timer expired
;  139     0247  1          CARD_IOSB_A:     VECTOR[4,WORD],                ! First card IOSB
;  140     0248  1          CARD_IOSB_B:     VECTOR[4,WORD],                ! Second card IOSB
;  141     0249  1          VALUE_DESC:      BBLOCK[DSC$C_D_BLN],           ! Qualifier value
;  142     0250  1          LOG_FILE_DESC:   BBLOCK[DSC$C_D_BLN],           ! /LOG_FILE descriptor
;  143     0251  1          NAME_DESC:       BBLOCK[DSC$C_D_BLN],           ! /NAME descriptor
;  144     0252  1          USERNAME_DESC:   BBLOCK[DSC$C_D_BLN],           ! Username descriptor
;  145     0253  1          PASSWORD_DESC:   BBLOCK[DSC$C_D_BLN],           ! Password descriptor
;  146     0254  1          CURRENT_COMMAND;                                ! Current command
;  147     0255  1
;  148     0256  1
;  149     0257  1 LITERAL
;  150     0258  1          K_NONE=          0,                             ! No significant command
;  151     0259  1          K_JOB=           1,                             ! JOB command
;  152     0260  1          K_EOJ=           3,                             ! EOJ command
;  153     0261  1          K_PASSWORD=      5;                             ! PASSWORD command
;  154     0262  1
;  155     0263  1
;  156     0264  1 LITERAL
;  157     0265  1          K_EFN_A=         1,                             ! EFN for first buffer
;  158     0266  1          K_EFN_B=         2;                             ! EFN for second buffer
;  159     0267  1
;  160     0268  1
;  161     0269  1 MACRO
;  162     0270  1          V_NO_LOG_FILE=           0,0,1,0 %,             ! /NOLOG specified
;  163     0271  1          V_SECOND_BUFFER=         0,1,1,0 %,             ! Second buffer has the read
;  164     0272  1          V_TRAILING_BLANKS=       0,2,1,0 %;             ! Leave trailing blanks
;  165     0273  1
;  166     0274  1
;  167     0275  1 BIND
;  168     0276  1          PERIODIC_INTERVAL = UPLIT(-150000000, -1);      ! 15 seconds
;  169     0277  1
;  170     0278  1
;  171     0279  1 FORWARD
```

```
;  172          0280  1          DOLLAR_STATES:          VECTOR[0],
;  173          0281  1          DOLLAR_KEYS:            VECTOR[0],
;  174          0282  1          JOB_STATES:             VECTOR[0],
;  175          0283  1          JOB_KEYS:               VECTOR[0],
;  176          0284  1          EOJ_STATES:             VECTOR[0],
;  177          0285  1          EOJ_KEYS:               VECTOR[0],
;  178          0286  1          PASSWORD_STATES:        VECTOR[0],
;  179          0287  1          PASSWORD_KEYS:          VECTOR[0];
;  180          0288  1
;  181          0289  1
;  182          0290  1 MACRO
;  183        M 0291  1          SD[A]=
;  184          0292  1              BIND %NAME('D_', A) = $DESCRIPTOR(A) %;
;  185          0293  1
;  186          0294  1
;  187        P 0295  1 SD(
;  188        P 0296  1          'P1',
;  189        P 0297  1          'AFTER',
;  190        P 0298  1          'CHARACTERISTICS',
;  191        P 0299  1          'CLI',
;  192        P 0300  1          'CPUTIME',
;  193        P 0301  1          'DELETE',
;  194        P 0302  1          'HOLD',
;  195        P 0303  1          'KEEP',
;  196        P 0304  1          'LOG_FILE',
;  197        P 0305  1          'NAME',
;  198        P 0306  1          'NOTIFY',
;  199        P 0307  1          'PARAMETERS',
;  200        P 0308  1          'PRINTER',
;  201        P 0309  1          'PRIORITY',
;  202        P 0310  1          'QUEUE',
;  203        P 0311  1          'RESTART',
;  204        P 0312  1          'TRAILING_BLANKS',
;  205        P 0313  1          'WSDEFAULT',
;  206        P 0314  1          'WSEXTENT',
;  207          0315  1          'WSQUOTA');
;  208          0316  1
;  209          0317  1
;  210          0318  1 BUILTIN
;  211          0319  1          MOVTC,
;  212          0320  1          TESTBITCC;
```

```
;  214     0321  1 ROUTINE INPSMB=
;  215     0322  1
;  216     0323  1 !++
;  217     0324  1 !
;  218     0325  1 ! FUNCTIONAL DESCRIPTION:
;  219     0326  1 !       This routine is the main entry point for the input symbiont.
;  220     0327  1 !
;  221     0328  1 ! INPUT PARAMETERS:
;  222     0329  1 !       Standard activation parameters (not used).
;  223     0330  1 !
;  224     0331  1 ! IMPLICIT INPUTS:
;  225     0332  1 !       NONE
;  226     0333  1 !
;  227     0334  1 ! OUTPUT PARAMETERS:
;  228     0335  1 !       NONE
;  229     0336  1 !
;  230     0337  1 ! IMPLICIT OUTPUTS:
;  231     0338  1 !       NONE
;  232     0339  1 !
;  233     0340  1 ! ROUTINE VALUE:
;  234     0341  1 !       Completion status.
;  235     0342  1 !
;  236     0343  1 ! SIDE EFFECTS:
;  237     0344  1 !       NONE
;  238     0345  1 !
;  239     0346  1 !--
;  240     0347  1
;  241     0348  2 BEGIN
;  242     0349  2 LOCAL
;  243     0350  2         DEVCLASS,                               ! Device class
;  244     0351  2         RSA_DESC:       VECTOR[2],              ! Descriptor for RSA
;  245     0352  2         DVI_DESC:       VECTOR[2],              ! Descriptor for DVI
;  246     0353  2         GETDVI_LIST:    BBLOCK[28],             ! $GETDVI item list
;  247     0354  2         IOSB:           VECTOR[4,WORD],         ! I/O status block
;  248     0355  2         STATUS_1,                               ! Status return
;  249     0356  2         STATUS_2,                               ! Status return
;  250     0357  2         STATUS_3;                               ! Status return
;  251     0358  2 BIND
;  252     0359  2         DEVICE_NAME = $DESCRIPTOR('SYS$INPUT:'): BBLOCK;
;  253     0360  2 BUILTIN
;  254     0361  2         FP;
;  255     0362  2
;  256     0363  2
;  257     0364  2 ! Establish the condition handler.
;  258     0365  2 !
;  259     0366  2 .FP = MAIN_HANDLER;
;  260     0367  2
;  261     0368  2
;  262     0369  2 ! Initialize RMS structures for the input stream.
;  263     0370  2 !
;  264   P 0371  2 $FAB_INIT(FAB=INPUT_FAB,
;  265   P 0372  2     FAC=GET,
;  266   P 0373  2     FNA=UPLIT BYTE('SYS$INPUT:'),
;  267   P 0374  2     FNS=%CHARCOUNT('SYS$INPUT:'),
;  268   P 0375  2     FOP=SQO,
;  269     0376  2     NAM=INPUT_NAM);
;  270   P 0377  2 $RAB_INIT(RAB=INPUT_RAB,
```

```
;  271        P 0378  2      FAB=INPUT_FAB,
;  272        P 0379  2      ROP=RAH,
;  273        P 0380  2      UBF=INPUT_UBF,
;  274          0381  2      USZ=80);
;  275        P 0382  2  $NAM_INIT(NAM=INPUT_NAM,
;  276        P 0383  2      ESA=INPUT_RSA,
;  277        P 0384  2      ESS=NAM$C_MAXRSS,
;  278        P 0385  2      RSA=INPUT_RSA,
;  279          0386  2      RSS=NAM$C_MAXRSS);
;  280          0387  2
;  281          0388  2
;  282          0389  2  ! Get the physical device name of the input device.
;  283          0390  2  !
;  284          0391  2  $PARSE(FAB=INPUT_FAB);
;  285          0392  2  DVI_DESC[0] = CH$RCHAR(INPUT_NAM[NAM$T_DVI]);
;  286          0393  2  DVI_DESC[1] = INPUT_NAM[NAM$T_DVI]+1;
;  287          0394  2  RSA_DESC[0] = 0;
;  288          0395  2  RSA_DESC[1] = INPUT_RSA;
;  289          0396  2
;  290          0397  2
;  291          0398  2  ! Execute a $GETDVI on the physical device.
;  292          0399  2  !
;  293          0400  2  GETDVI_LIST[0,0,16,0] = 4;
;  294          0401  2  GETDVI_LIST[2,0,16,0] = DVI$_DEVCLASS;
;  295          0402  2  GETDVI_LIST[4,0,32,0] = DEVCLASS;
;  296          0403  2  GETDVI_LIST[8,0,32,0] = 0;
;  297          0404  2  GETDVI_LIST[12,0,16,0] = NAM$C_MAXRSS;
;  298          0405  2  GETDVI_LIST[14,0,16,0] = DVI$_DEVNAM;
;  299          0406  2  GETDVI_LIST[16,0,32,0] = INPUT_RSA;
;  300          0407  2  GETDVI_LIST[20,0,32,0] = RSA_DESC;
;  301          0408  2  GETDVI_LIST[24,0,32,0] = 0;
;  302        P 0409  2  STATUS_1 = $GETDVIW(
;  303        P 0410  2      IOSB=IOSB,
;  304        P 0411  2      DEVNAM=DVI_DESC,
;  305          0412  2      ITMLST=GETDVI_LIST);
;  306          0413  2  IF NOT .STATUS_1 THEN RETURN .STATUS_1;
;  307          0414  2
;  308          0415  2
;  309          0416  2  ! Open the input stream.
;  310          0417  2  !
;  311          0418  2  IF .DEVCLASS EQL DC$_CARD
;  312          0419  2  THEN
;  313          0420  3      BEGIN
;  314          0421  3
;  315          0422  3      ! Set up to issue signalled messages to the card operator.
;  316          0423  3      !
;  317          0424  3      PUTMSG_ACTION_ROUTINE = MAIN_HANDLER_ACTION;
;  318          0425  3
;  319          0426  3
;  320          0427  3      ! Open the card reader.
;  321          0428  3      !
;  322          0429  3      STATUS_2 = $ASSIGN(DEVNAM=DEVICE_NAME, CHAN=CARD_CHANNEL);
;  323          0430  3      IF NOT .STATUS_2
;  324          0431  3      THEN
;  325          0432  3          SIGNAL(
;  326          0433  3              INPSMB$_FACILITY^16 + SHR$_OPENIN + STS$K_SEVERE,
;  327          0434  3              1, RSA_DESC,
```

```
;  328     0435  3                .STATUS_2);
;  329     0436  3          INPUT_NAM[NAM$B_RSL] = .RSA_DESC[0];
;  330     0437
;  331     0438
;  332     0439  3          ! Set up the periodic timer.
;  333     0440  3          !
;  334     0441  3          $SETIMR(DAYTIM=PERIODIC_INTERVAL, ASTADR=TIMER_AST);
;  335     0442
;  336     0443
;  337     0444  3          ! Start a read in the first buffer.
;  338     0445  3          !
;  339   P 0446  3          STATUS_3 = $QIO(
;  340   P 0447  3              EFN=K_EFN_A,
;  341   P 0448  3              FUNC=IO$_READLBLK,
;  342   P 0449  3              CHAN=.CARD_CHANNEL,
;  343   P 0450  3              IOSB=CARD_IOSB_A,
;  344   P 0451  3              P1=INPUT_OBF,
;  345     0452  3              P2=80);
;  346     0453  3          IF NOT .STATUS_3
;  347     0454  3          THEN
;  348     0455  3              FILE_ERROR(
;  349     0456  3                  INPSMB$_FACILITY^16 + SHR$_READERR + STS$K_SEVERE,
;  350     0457  3                  INPUT_FAB,
;  351     0458  3                  .STATUS_3);
;  352     0459  3          END
;  353     0460  2      ELSE
;  354     0461  2          BEGIN
;  355     0462  2
;  356     0463  2          ! Access the file with RMS.
;  357     0464  2          !
;  358     0465  2          IF NOT $OPEN(FAB=INPUT_FAB)
;  359     0466  2          THEN
;  360     0467  2              FILE_ERROR(
;  361     0468  2                  INPSMB$_FACILITY^16 + SHR$_OPENIN + STS$K_SEVERE,
;  362     0469  2                  INPUT_FAB,
;  363     0470  2                  .INPUT_FAB[FAB$L_STS], .INPUT_FAB[FAB$L_STV]);
;  364     0471  2
;  365     0472  2
;  366     0473  2          IF NOT $CONNECT(RAB=INPUT_RAB)
;  367     0474  2          THEN
;  368     0475  2              FILE_ERROR(
;  369     0476  2                  INPSMB$_FACILITY^16 + SHR$_OPENIN + STS$K_SEVERE,
;  370     0477  2                  INPUT_FAB,
;  371     0478  2                  .INPUT_RAB[RAB$L_STS], .INPUT_RAB[RAB$L_STV]);
;  372     0479  2          END;
;  373     0480  2
;  374     0481  2
;  375     0482  2      ! Initialize descriptors for dynamic strings.
;  376     0483  2      !
;  377     0484  2      VALUE_DESC[DSC$B_CLASS] = DSC$K_CLASS_D;
;  378     0485  2      VALUE_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  379     0486  2      VALUE_DESC[DSC$W_LENGTH] = 0;
;  380     0487  2      VALUE_DESC[DSC$A_POINTER] = 0;
;  381     0488  2
;  382     0489  2      LOG_FILE_DESC[DSC$B_CLASS] = DSC$K_CLASS_D;
;  383     0490  2      LOG_FILE_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  384     0491  2      LOG_FILE_DESC[DSC$W_LENGTH] = 0;
```

```
;  385          0492  2 LOG_FILE_DESC[DSC$A_POINTER] = 0;
;  386          0493  2
;  387          0494  2 NAME_DESC[DSC$B_CLASS] = DSC$K_CLASS_D;
;  388          0495  2 NAME_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  389          0496  2 NAME_DESC[DSC$W_LENGTH] = 0;
;  390          0497  2 NAME_DESC[DSC$A_POINTER] = 0;
;  391          0498  2
;  392          0499  2 USERNAME_DESC[DSC$B_CLASS] = DSC$K_CLASS_D;
;  393          0500  2 USERNAME_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  394          0501  2 USERNAME_DESC[DSC$W_LENGTH] = 0;
;  395          0502  2 USERNAME_DESC[DSC$A_POINTER] = 0;
;  396          0503  2
;  397          0504  2 PASSWORD_DESC[DSC$B_CLASS] = DSC$K_CLASS_D;
;  398          0505  2 PASSWORD_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  399          0506  2 PASSWORD_DESC[DSC$W_LENGTH] = 0;
;  400          0507  2 PASSWORD_DESC[DSC$A_POINTER] = 0;
;  401          0508  2
;  402          0509  2
;  403          0510  2 ! Loop to process all jobs in the input stream.
;  404          0511  2 !
;  405          0512  2 UNTIL PROCESSING_LOOP() DO 0;
;  406          0513  2
;  407          0514  2
;  408          0515  2 ! Close the input stream.
;  409          0516  2 !
;  410          0517  2 IF .CARD_CHANNEL EQL 0
;  411          0518  2 THEN
;  412          0519  3     IF NOT $CLOSE(FAB=INPUT_FAB)
;  413          0520  3     THEN
;  414          0521  3         FILE_ERROR(
;  415          0522  3             INPSMB$_FACILITY^16 + SHR$_CLOSEIN + STS$K_SEVERE,
;  416          0523  3             INPUT_FAB,
;  417          0524  3             .INPUT_FAB[FAB$L_STS], .INPUT_FAB[FAB$L_STV]);
;  418          0525  2
;  419          0526  2
;  420          0527  2 ! Exit the symbiont.
;  421          0528  2 !
;  422          0529  2 SS$_NORMAL
;  423          0530  1 END;
; INFO#250                 L1:0418
; Referenced LOCAL symbol DEVCLASS is probably not initialized

                                .TITLE  INPSMB Input symbiont
                                .IDENT  \V04-000\

                                .PSECT  DATA,NOEXE,2

00000 CARD_CHANNEL:
                                .BLKB   2
00002                           .BLKB   2
00004 INPUT_FAB:
                                .BLKB   80
00054 INPUT_RAB:
                                .BLKB   68
00098 INPUT_NAM:
                                .BLKB   96
```

```
000F8 INPUT_RSA:
                .BLKB   255
001F7           .BLKB   1
001F8 INPUT_UBF:
                .BLKB   160
00298 OUTPUT_FAB:
                .BLKB   80
002E8 OUTPUT_RAB:
                .BLKB   68
0032C OUTPUT_NAM:
                .BLKB   96
0038C OUTPUT_XAB:
                .BLKB   88
003E4 OUTPUT_RSA:
                .BLKB   255
004E3           .BLKB   1
004E4 JOB_LENGTH:
                .BLKB   4
004E8 JOB_BUFFER:
                .BLKB   80
00538 PUTMSG_ACTION_ROUTINE:
                .BLKB   4
0053C FLAGS:    .BLKB   4
00540 INPUT_COMPLETIONS:
                .BLKB   4
00544 CARD_IOSB_A:
                .BLKB   8
0054C CARD_IOSB_B:
                .BLKB   8
00554 VALUE_DESC:
                .BLKB   8
0055C LOG_FILE_DESC:
                .BLKB   8
00564 NAME_DESC:
                .BLKB   8
0056C USERNAME_DESC:
                .BLKB   8
00574 PASSWORD_DESC:
                .BLKB   8
0057C CURRENT_COMMAND:
                .BLKB   4

                .PSECT  CODE,NOWRT,2

                FFFFFFFF  F70F2E80  00000 P.AAA:  .LONG   -150000000, -1                 ;
                            31  50  00008 P.AAC:  .ASCII  \P1\                          ;
                                    0000A         .BLKB   2
                          00000002  0000C P.AAB:  .LONG   2                             ;
                          00000000' 00010         .ADDRESS P.AAC
                      52  45  54  46  41  00014 P.AAE:  .ASCII  \AFTER\                  ;
                                    00019         .BLKB   3
                          00000005  0001C P.AAD:  .LONG   5                             ;
                          00000000' 00020         .ADDRESS P.AAE
53  43  49  54  53  49  52  45  54  43  41  52  41  48  43  00024 P.AAG:  .ASCII  \CHARACTERISTICS\  ;
                                    00033         .BLKB   1
                          0000000F  00034 P.AAF:  .LONG   15                            ;
                          00000000' 00038         .ADDRESS P.AAG
```

```
                                                49  4C  43  0003C P.AAI:   .ASCII  \CLI\                                ;
                                                            0003F         .BLKB   1
                                              00000003      00040 P.AAH:  .LONG   3                                      ;
                                              00000000'     00044         .ADDRESS P.AAI                                 ;
                                        45  4D  49  54  55  50  43  00048 P.AAK:  .ASCII  \CPUTIME\                      ;
                                                            0004F         .BLKB   1
                                              00000007      00050 P.AAJ:  .LONG   7                                      ;
                                              00000000'     00054         .ADDRESS P.AAK                                 ;
                                            45  54  45  4C  45  44  00058 P.AAM:  .ASCII  \DELETE\                      ;
                                                            0005E         .BLKB   2
                                              00000006      00060 P.AAL:  .LONG   6                                      ;
                                              00000000'     00064         .ADDRESS P.AAM                                 ;
                                                44  4C  4F  48  00068 P.AAO:  .ASCII  \HOLD\                         ;
                                              00000004      0006C P.AAN:  .LONG   4                                      ;
                                              00000000'     00070         .ADDRESS P.AAO                                 ;
                                                50  45  45  4B  00074 P.AAQ:  .ASCII  \KEEP\                         ;
                                              00000004      00078 P.AAP:  .LONG   4                                      ;
                                              00000000'     0007C         .ADDRESS P.AAQ                                 ;
                                        45  4C  49  46  5F  47  4F  4C  00080 P.AAS:  .ASCII  \LOG_FILE\                 ;
                                              00000008      00088 P.AAR:  .LONG   8                                      ;
                                              00000000'     0008C         .ADDRESS P.AAS                                 ;
                                                45  4D  41  4E  00090 P.AAU:  .ASCII  \NAME\                         ;
                                              00000004      00094 P.AAT:  .LONG   4                                      ;
                                              00000000'     00098         .ADDRESS P.AAU                                 ;
                                            59  46  49  54  4F  4E  0009C P.AAW:  .ASCII  \NOTIFY\                      ;
                                                            000A2         .BLKB   2
                                              00000006      000A4 P.AAV:  .LONG   6                                      ;
                                              00000000'     000A8         .ADDRESS P.AAW                                 ;
                                53  52  45  54  45  4D  41  52  41  50  000AC P.AAY:  .ASCII  \PARAMETERS\              ;
                                                            000B6         .BLKB   2
                                                0000000A      000B8 P.AAX:  .LONG   10                                     ;
                                                00000000'     000BC         .ADDRESS P.AAY                                 ;
                                                    52  45  54  4E  49  52  50  000C0 P.ABA:  .ASCII  \PRINTER\          ;
                                                            000C7         .BLKB   1
                                                00000007      000C8 P.AAZ:  .LONG   7                                      ;
                                                00000000'     000CC         .ADDRESS P.ABA                                 ;
                                                59  54  49  52  4F  49  52  50  000D0 P.ABC:  .ASCII  \PRIORITY\         ;
                                                00000008      000D8 P.ABB:  .LONG   8                                      ;
                                                00000000'     000DC         .ADDRESS P.ABC                                 ;
                                                        45  55  45  55  51  000E0 P.ABE:  .ASCII  \QUEUE\                ;
                                                            000E5         .BLKB   3
                                                00000005      000E8 P.ABD:  .LONG   5                                      ;
                                                00000000'     000EC         .ADDRESS P.ABE                                 ;
                                                    54  52  41  54  53  45  52  000F0 P.ABG:  .ASCII  \RESTART\          ;
                                                            000F7         .BLKB   1
                                                00000007      000F8 P.ABF:  .LONG   7                                      ;
                                                00000000'     000FC         .ADDRESS P.ABG                                 ;
53  4B  4E  41  4C  42  5F  47  4E  49  4C  49  41  52  54  00100 P.ABI:  .ASCII  \TRAILING_BLANKS\                       ;
                                                            0010F         .BLKB   1
                                                0000000F      00110 P.ABH:  .LONG   15                                     ;
                                                00000000'     00114         .ADDRESS P.ABI                                 ;
                                                    54  4C  55  41  46  45  44  53  57  00118 P.ABK:  .ASCII  \WSDEFAULT\  ;
                                                            00121         .BLKB   3
                                                00000009      00124 P.ABJ:  .LONG   9                                      ;
                                                00000000'     00128         .ADDRESS P.ABK                                 ;
                                                    54  4E  45  54  58  45  53  57  0012C P.ABM:  .ASCII  \WSEXTENT\      ;
                                                00000008      00134 P.ABL:  .LONG   8                                      ;
```

```
                                00000000' 00138          .ADDRESS P.ABM                 ;
         41 54 4F 55 51 53 57   0013C P.ABO:   .ASCII   \WSQUOTA\                       ;
                                00143          .BLKB    1
                                00000007  00144 P.ABN:   .LONG    7                     ;
                                00000000' 00148          .ADDRESS P.ABO                 ;
   3A 54 55 50 4E 49 24 53 59 53   0014C P.ABQ:   .ASCII   \SYS$INPUT:\                 ;
                                00156          .BLKB    2
                                0000000A  00158 P.ABP:   .LONG    10                    ;
                                00000000' 0015C          .ADDRESS P.ABQ                 ;
   3A 54 55 50 4E 49 24 53 59 53   00160 P.ABR:   .ASCII   \SYS$INPUT:\                 ;

                                PERIODIC_INTERVAL=  P.AAA
                                D_P1=               P.AAB
                                D_AFTER=            P.AAD
                                D_CHARACTERISTICS=  P.AAF
                                D_CLI=              P.AAH
                                D_CPUTIME=          P.AAJ
                                D_DELETE=           P.AAL
                                D_HOLD=             P.AAN
                                D_KEEP=             P.AAP
                                D_LOG_FILE=         P.AAR
                                D_NAME=             P.AAT
                                D_NOTIFY=           P.AAV
                                D_PARAMETERS=       P.AAX
                                D_PRINTER=          P.AAZ
                                D_PRIORITY=         P.ABB
                                D_QUEUE=            P.ABD
                                D_RESTART=          P.ABF
                                D_TRAILING_BLANKS=  P.ABH
                                D_WSDEFAULT=        P.ABJ
                                D_WSEXTENT=         P.ABL
                                D_WSQUOTA=          P.ABN
                                DEVICE_NAME=        P.ABP
                                $RMS_PTR=           INPUT_FAB
                                $RMS_PTR=           INPUT_RAB
                                $RMS_PTR=           INPUT_NAM
                                         .EXTRN   CLI$DCL_PARSE, CLI$GET_VALUE
                                         .EXTRN   CLI$PRESENT, LGI$VALIDATE
                                         .EXTRN   LIB$SFREE1_DD, LIB$SIGNAL
                                         .EXTRN   LIB$TPARSE, LIB$AB_UPCASE
                                         .EXTRN   INPSMBCLD, INPSMB$_FACILITY
                                         .EXTRN   INPSMB$_ENTFIL, INPSMB$_INVCONT
                                         .EXTRN   INPSMB$_INVLOGFIL
                                         .EXTRN   INPSMB$_INVPASS
                                         .EXTRN   INPSMB$_INVUSER
                                         .EXTRN   INPSMB$_JOBCARD
                                         .EXTRN   INPSMB$_MISSPASS
                                         .EXTRN   INPSMB$_OPENUAF
                                         .EXTRN   INPSMB$_USERVAL
                                         .EXTRN   SYS$PARSE, SYS$GETDVIW
                                         .EXTRN   SYS$ASSIGN, SYS$SETIMR
                                         .EXTRN   SYS$QIO, SYS$OPEN
                                         .EXTRN   SYS$CONNECT, SYS$CLOSE

                          00FC 00000 INPSMB: .WORD    Save R2,R3,R4,R5,R6,R7          ; 0321
         57      0000V  CF   9E 00002          MOVAB    FILE_ERROR, R7                  ;
         56      0000'  CF   9E 00007          MOVAB    $RMS_PTR, R6                    ;
```

```
                        5E                  38  C2 0000C       SUBL2   #56, SP
                        6D          0000V   CF  9E 0000F       MOVAB   MAIN_HANDLER, (FP)                                      ;0366
0050    8F      00      6E                  00  2C 00014       MOVC5   #0, (SP), #0, #80, $RMS_PTR                             ;0376
                                            66     0001B
                        66          5003    8F  B0 0001C       MOVW    #20483, $RMS_PTR
                    04  A6            40    8F  9A 00021       MOVZBL  #64, $RMS_PTR+4
                    16  A6                  02  90 00026       MOVB    #2, $RMS_PTR+22
                    1F  A6                  02  90 0002A       MOVB    #2, $RMS_PTR+31
                    28  A6          0094    C6  9E 0002E       MOVAB   INPUT_NAM, $RMS_PTR+40
                    2C  A6            BF    AF  9E 00034       MOVAB   P.ABR, $RMS_PTR+44
                    34  A6                  0A  90 00039       MOVB    #10, $RMS_PTR+52
0044    8F      00      6E                  00  2C 0003D       MOVC5   #0, (SP), #0, #68, $RMS_PTR                             ;0381
                                    50      A6     00044
                    50  A6          4401    8F  B0 00046       MOVW    #17409, $RMS_PTR
                    54  A6          0200    8F  3C 0004C       MOVZWL  #512, $RMS_PTR+4
                    70  A6            50    8F  9B 00052       MOVZBW  #80, $RMS_PTR+32
                    74  A6          01F4    C6  9E 00057       MOVAB   INPUT_UBF, $RMS_PTR+36
                008C    C6                  66  9E 0005D       MOVAB   INPUT_FAB, $RMS_PTR+60
0060    8F      00      6E                  00  2C 00062       MOVC5   #0, (SP), #0, #96, $RMS_PTR                             ;0386
                                    0094    C6     00069
                0094    C6          6002    8F  B0 0006C       MOVW    #24578, $RMS_PTR
                0096    C6                  01  8E 00073       MNEGB   #1, $RMS_PTR+2
                0098    C6          00F4    C6  9E 00078       MOVAB   INPUT_RSA, $RMS_PTR+4
                009E    C6                  01  8E 0007F       MNEGB   #1, $RMS_PTR+10
                00A0    C6          00F4    C6  9E 00084       MOVAB   INPUT_RSA, $RMS_PTR+12
                                            56  DD 00088       PUSHL   R6                                                      ;0391
        00000000G       00                  01  FB 0008D       CALLS   #1, SYS$PARSE
                  28    AE          00A8    C6  9A 00094       MOVZBL  INPUT_NAM+20, DVI_DESC                                  ;0392
                  2C    AE          00A9    C6  9E 0009A       MOVAB   INPUT_NAM+21, DVI_DESC+4                                ;0393
                                      30    AE  D4 000A0       CLRL    RSA_DESC                                                ;0394
                  34    AE          00F4    C6  9E 000A3       MOVAB   INPUT_RSA, RSA_DESC+4                                   ;0395
                  0C    AE    00040004      8F  D0 000A9       MOVL    #262148, GETDVI_LIST                                    ;0400
                  10    AE                  6E  9E 000B1       MOVAB   DEVCLASS, GETDVI_LIST+4                                 ;0402
                                      14    AE  D4 000B5       CLRL    GETDVI_LIST+8                                           ;0403
                  18    AE    002000FF      8F  D0 000B8       MOVL    #2097407, GETDVI_LIST+12                                ;0404
                  1C    AE          00F4    C6  9E 000C0       MOVAB   INPUT_RSA, GETDVI_LIST+16                               ;0406
                  20    AE            30    AE  9E 000C6       MOVAB   RSA_DESC, GETDVI_LIST+20                                ;0407
                                      24    AE  D4 000CB       CLRL    GETDVI_LIST+24                                          ;0408
                                            7E  7C 000CE       CLRQ    -(SP)                                                   ;0412
                                            7E  D4 000D0       CLRL    -(SP)
                                      10    AE  9F 000D2       PUSHAB  IOSB
                                      1C    AE  9F 000D5       PUSHAB  GETDVI_LIST
                                      3C    AE  9F 000D8       PUSHAB  DVI_DESC
                                            7E  7C 000DB       CLRQ    -(SP)
        00000000G       00                  08  FB 000DD       CALLS   #8, SYS$GETDVIW
                        01                  50  E8 000E4       BLBS    STATUS_1, 1$                                            ;0413
                                            04  000E7       RET
        00000041        8F                  6E  D1 000E8 1$:   CMPL    DEVCLASS, #65                                           ;0418
                                            7A  12 000EF       BNEQ    3$
                0534    C6          0000V   CF  9E 000F1       MOVAB   MAIN_HANDLER_ACTION, PUTMSG_ACTION_ROUTINE              ;0424
                                            7E  7C 000F8       CLRQ    -(SP)                                                   ;0429
                                        FC  A6  9F 000FA       PUSHAB  CARD_CHANNEL
                                      FEED  CF  9F 000FD       PUSHAB  DEVICE_NAME
        00000000G       00                  04  FB 00101       CALLS   #4, SYS$ASSIGN
                        14                  50  E8 00108       BLBS    STATUS_2, 2$                                            ;0430
                                            50  DD 0010B       PUSHL   STATUS_2                                                ;0435
                                      34    AE  9F 0010D       PUSHAB  RSA_DESC                                                ;0432
```

```
                                01 DD 00110          PUSHL   #1                                                              ;
                      00000000* 8F DD 00112          PUSHL   #<<<INPSMB$_FACILITY@16>+4248>+4>                               ; 0433
00000000G   00                  04 FB 00118          CALLS   #4, LIB$SIGNAL
      0097  C6              30  AE 90 0011F 2$:      MOVB    RSA_DESC, INPUT_NAM+3                                           ; 0436
                                7E D4 00125          CLRL    -(SP)                                                           ; 0441
                          0000V CF 9F 00127          PUSHAB  TIMER_AST
                          FD67  CF 9F 0012B          PUSHAB  PERIODIC_INTERVAL
                                7E D4 0012F          CLRL    -(SP)
00000000G   00                  04 FB 00131          CALLS   #4, SYS$SETIMR
                                7E 7C 00138          CLRQ    -(SP)                                                           ; 0452
                                7E 7C 0013A          CLRQ    -(SP)
                      7E        50 8F 9A 0013C       MOVZBL  #80, -(SP)
                          01F4  C6 9F 00140          PUSHAB  INPUT_UBF
                                7E 7C 00144          CLRQ    -(SP)
                          0540  C6 9F 00146          PUSHAB  CARD_IOSB_A
                                21 DD 0014A          PUSHL   #33
                      7E        FC A6 3C 0014C       MOVZWL  CARD_CHANNEL, -(SP)
                                01 DD 00150          PUSHL   #1
00000000G   00                  0C FB 00152          CALLS   #12, SYS$QIO
                        46      50 E8 00159          BLBS    STATUS_3, 5$                                                    ; 0453
                                50 DD 0015C          PUSHL   STATUS_3                                                        ; 0458
                                56 DD 0015E          PUSHL   R6                                                              ; 0455
                      0000C0000* 8F DD 00160         PUSHL   #<<<INPSMB$_FACILITY@16>+4272>+4>                               ; 0456
                        67      03 FB 00166          CALLS   #3, FILE_ERROR
                                37 11 00169          BRB     5$                                                              ; 0418
                                56 DD 0016B 3$:      PUSHL   R6                                                              ; 0465
00000000G   00                  01 FB 0016D          CALLS   #1, SYS$OPEN
                        0F      50 E8 00174          BLBS    R0, 4$                                                          ;
                      7E        08 A6 7D 00177       MOVQ    INPUT_FAB+8, -(SP)                                              ; 0470
                                56 DD 0017B          PUSHL   R6                                                              ; 0467
                      00000000* 8F DD 0017D          PUSHL   #<<<INPSMB$_FACILITY@16>+4248>+4>                               ; 0468
                        67      04 FB 00183          CALLS   #4, FILE_ERROR
                              50 A6 9F 00186 4$:      PUSHAB  INPUT_RAB                                                       ; 0473
00000000G   00                  01 FB 00189          CALLS   #1, SYS$CONNECT
                        0F      50 E8 00190          BLBS    R0, 5$                                                          ;
                      7E        58 A6 7D 00193       MOVQ    INPUT_RAB+8, -(SP)                                              ; 0478
                                56 DD 00197          PUSHL   R6                                                              ; 0475
                      00000000* 8F DD 00199          PUSHL   #<<<INPSMB$_FACILITY@16>+4248>+4>                               ; 0476
                        67      04 FB 0019F          CALLS   #4, FILE_ERROR
          0550  C6  020E0000    8F D0 001A2 5$:      MOVL    #34471936, VALUE_DESC                                           ; 0486
                          0554  C6 D4 001AB          CLRL    VALUE_DESC+4                                                    ; 0487
          0558  C6  020E0000    8F D0 001AF          MOVL    #34471936, LOG_FILE_DESC                                        ; 0491
                          055C  C6 D4 001B8          CLRL    LOG_FILE_DESC+4                                                 ; 0492
          0560  C6  020E0000    8F D0 001BC          MOVL    #34471936, NAME_DESC                                            ; 0496
                          0564  C6 D4 001C5          CLRL    NAME_DESC+4                                                     ; 0497
          0568  C6  020E0000    8F D0 001C9          MOVL    #34471936, USERNAME_DESC                                        ; 0501
                          056C  C6 D4 001D2          CLRL    USERNAME_DESC+4                                                 ; 0502
          0570  C6  020E0000    8F D0 001D6          MOVL    #34471936, PASSWORD_DESC                                        ; 0506
                          0574  C6 D4 001DF          CLRL    PASSWORD_DESC+4                                                 ; 0507
          0000V CF              00 FB 001E3 6$:      CALLS   #0, PROCESSING_LOOP                                             ; 0512
                        FB      50 E9 001E8          BLBC    R0, 6$
                              FC A6 B5 001EB         TSTW    CARD_CHANNEL                                                    ; 0517
                                1B 12 001EE          BNEQ    7$
                                56 DD 001F0          PUSHL   R6                                                              ; 0519
00000000G   00                  01 FB 001F2          CALLS   #1, SYS$CLOSE
                        0F      50 E8 001F9          BLBS    R0, 7$                                                          ;
                      7E        08 A6 7D 001FC       MOVQ    INPUT_FAB+8, -(SP)                                              ; 0524
```

```
                        56  DD 00200        PUSHL   R6                                              ; 0521
              00000000*  8F  DD 00202        PUSHL   #<<<INPSMB$_FACILITY@16>+4176>+4>               ; 0522
                     67  04  FB 00208        CALLS   #4, FILE_ERROR                                  ;
                     50  01  D0 0020B 7$:     MOVL    #1, R0                                          ; 0530
                         04 0020E            RET                                                     ;

; Routine Size:  527 bytes,    Routine Base:  CODE + 016A
```

```
;  425     0531  1 ROUTINE PROCESSING_LOOP_HANDLER(SIG,MCH)=
;  426     0532  1
;  427     0533  1 !++
;  428     0534  1 !
;  429     0535  1 ! FUNCTIONAL DESCRIPTION:
;  430     0536  1 !       This is a condition handler for routine PROCESSING_LOOP.
;  431     0537  1 !
;  432     0538  1 ! INPUT PARAMETERS:
;  433     0539  1 !       Standard VMS condition handler parameters.
;  434     0540  1 !
;  435     0541  1 ! IMPLICIT INPUTS:
;  436     0542  1 !       NONE
;  437     0543  1 !
;  438     0544  1 ! OUTPUT PARAMETERS:
;  439     0545  1 !       NONE
;  440     0546  1 !
;  441     0547  1 ! IMPLICIT OUTPUTS:
;  442     0548  1 !       NONE
;  443     0549  1 !
;  444     0550  1 ! ROUTINE VALUE:
;  445     0551  1 !       NONE
;  446     0552  1 !
;  447     0553  1 ! SIDE EFFECTS:
;  448     0554  1 !       NONE
;  449     0555  1 !
;  450     0556  1 !--
;  451     0557  1
;  452     0558  2 BEGIN
;  453     0559  2 MAP
;  454     0560  2         SIG:            REF BBLOCK,     ! Signal parameters
;  455     0561  2         MCH:            REF BBLOCK;     ! Mechanism parameters
;  456     0562  2 LOCAL
;  457     0563  2         COND:           BBLOCK[4];      ! Status value
;  458     0564  2 BUILTIN
;  459     0565  2         AP,
;  460     0566  2         CALLG;
;  461     0567  2
;  462     0568  2
;  463     0569  2 ! Get the condition that was signalled.
;  464     0570  2 !
;  465     0571  2 COND = .SIG[CHF$L_SIG_NAME];
;  466     0572  2
;  467     0573  2
;  468     0574  2 IF .COND NEQ SS$_UNWIND
;  469     0575  2 THEN
;  470     0576  2     BEGIN
;  471     0577  2
;  472     0578  2     ! Downgrade the severity of any message issued to error.
;  473     0579  2     !
;  474     0580  2     IF .COND[STS$V_SEVERITY] EQL STS$K_SEVERE
;  475     0581  2     THEN
;  476     0582  2         BBLOCK[SIG[CHF$L_SIG_NAME], STS$V_SEVERITY] = STS$K_ERROR;
;  477     0583  2
;  478     0584  2
;  479     0585  2     ! Call the main handler to issue the message.
;  480     0586  2     !
;  481     0587  2     CALLG(.AP, MAIN_HANDLER);
```

```
;  482          0588  3
;  483          0589  3
;  484          0590  3      ! If the message is an error status, clean up the current job.
;  485          0591  3      !
;  486          0592  3      IF NOT .COND
;  487          0593  3      THEN
;  488          0594  4          BEGIN
;  489          0595  4
;  490          0596  4          ! Close and delete the command procedure file if it is open.
;  491          0597  4          !
;  492          0598  4          IF .OUTPUT_FAB[FAB$W_IFI] NEQ 0
;  493          0599  4          THEN
;  494          0600  5              BEGIN
;  495          0601  5              OUTPUT_FAB[FAB$V_DLT] = TRUE;
;  496          0602  5              $CLOSE(FAB=OUTPUT_FAB);
;  497          0603  4              END;
;  498          0604  4
;  499          0605  4
;  500          0606  4          ! Unwind to the caller of PROCESSING_LOOP with a false value.
;  501          0607  4          !
;  502          0608  4          MCH[CHF$L_MCH_SAVR0] = FALSE;
;  503          0609  4          $UNWIND();
;  504          0610  3          END;
;  505          0611  2      END;
;  506          0612  2
;  507          0613  2
;  508          0614  2 SS$_CONTINUE
;  509          0615  1 END;


                                                .EXTRN  SYS$UNWIND

                                        0004 00000 PROCESSING_LOOP_HANDLER:
                                                   .WORD   Save R2                                  ; 0531
                            50      04  AC  D0 00002        MOVL    SIG, R0                         ; 0571
                            52      04  A0  D0 00006        MOVL    4(R0), COND
                  00000920  8F          52  D1 0000A        CMPL    COND, #2336                     ; 0574
                                        3C  13 00011        BEQL    3$
        04          52          03      00  ED 00013        CMPZV   #0, #3, COND, #4                ; 0580
                                        06  12 00018        BNEQ    1$
    04  A0          03          00      02  F0 0001A        INSV    #2, #0, #3, 4(R0)               ; 0582
                  0000V  CF             6C  FA 00020 1$:    CALLG   (AP), MAIN_HANDLER              ; 0587
                            27          52  E8 00025        BLBS    COND, 3$                        ; 0592
                                0000'   CF  B5 00028        TSTW    OUTPUT_FAB+2                    ; 0598
                                        11  13 0002C        BEQL    2$
                  0000'  CF     80      8F  88 0002E        BISB2   #128, OUTPUT_FAB+5              ; 0601
                                0000'   CF  9F 00034        PUSHAB  OUTPUT_FAB                      ; 0602
                  00000000G  00         01  FB 00036        CALLS   #1, SYS$CLOSE
                            50      08  AC  D0 0003F 2$:    MOVL    MCH, R0                         ; 0608
                                    0C  A0  D4 00043        CLRL    12(R0)
                                        7E  7C 00046        CLRQ    -(SP)                           ; 0609
                  00000000G  00         02  FB 00048        CALLS   #2, SYS$UNWIND
                            50          01  D0 0004F 3$:    MOVL    #1, R0                          ; 0615
                                        04  00052        RET

; Routine Size:  83 bytes,    Routine Base:  CODE + 0379
```

```
;  511     0616  1 ROUTINE PROCESSING_LOOP=
;  512     0617  1
;  513     0618  1 !++
;  514     0619  1 !
;  515     0620  1 ! FUNCTIONAL DESCRIPTION:
;  516     0621  1 !       This routine implements the main control sequencing for the input
;  517     0622  1 !       symbiont.
;  518     0623  1 !
;  519     0624  1 ! INPUT PARAMETERS:
;  520     0625  1 !       NONE
;  521     0626  1 !
;  522     0627  1 ! IMPLICIT INPUTS:
;  523     0628  1 !       NONE
;  524     0629  1 !
;  525     0630  1 ! OUTPUT PARAMETERS:
;  526     0631  1 !       NONE
;  527     0632  1 !
;  528     0633  1 ! IMPLICIT OUTPUTS:
;  529     0634  1 !       NONE
;  530     0635  1 !
;  531     0636  1 ! ROUTINE VALUE:
;  532     0637  1 !       NONE
;  533     0638  1 !
;  534     0639  1 ! SIDE EFFECTS:
;  535     0640  1 !       NONE
;  536     0641  1 !
;  537     0642  1 !--
;  538     0643  1
;  539     0644  2 BEGIN
;  540     0645  2 PARSE_GLOBAL_REGISTERS;
;  541     0646  2 LOCAL
;  542     0647  2         ITEM_BUFFER:    BBLOCK[2048],           ! $SNDJBC item buffer
;  543     0648  2         DATA_BUFFER:    BBLOCK[2048],           ! $SNDJBC data buffer
;  544     0649  2         UAF_BUFFER:     BBLOCK[UAF$C_LENGTH],   ! UAF record for user
;  545     0650  2         UAF_DESC:       VECTOR[2],              ! Descriptor for UAF buffer
;  546     0651  2         DNA_BUFFER:     VECTOR[NAM$C_MAXRSS,BYTE], ! Default filename
;  547     0652  2         DNA_DESC:       VECTOR[2],              ! Descriptor for DNA buffer
;  548     0653  2         IOSB:           VECTOR[4,WORD],         ! $SNDJBC status block
;  549     0654  2         LINE_DESC:      BBLOCK[DSC$C_S_BLN],    ! Descriptor for command
;  550     0655  2         STATUS_1,                               ! Status return
;  551     0656  2         STATUS_2;                               ! Status return
;  552     0657  2 BUILTIN
;  553     0658  2         FP;
;  554     0659  2
;  555     0660  2
;  556     0661  2 ! Establish the condition handler.
;  557     0662  2 !
;  558     0663  2 .FP = PROCESSING_LOOP_HANDLER;
;  559     0664  2
;  560     0665  2
;  561     0666  2 ! Initialize for command parsing utilities.
;  562     0667  2 !
;  563   P 0668  2 PARSE_GLOBAL_INIT(
;  564   P 0669  2     ICURSOR=    ITEM_BUFFER,
;  565   P 0670  2     DCURSOR=    DATA_BUFFER,
;  566   P 0671  2     MESSAGE=    INPSMB$_FACILITY^16 OR SHR$_BADQNAME OR STS$K_SEVERE,
;  567     0672  2     VALUE_DESC= VALUE_DESC);
```

```
;  568          0673  2
;  569          0674  2
;  570          0675  2 ! Read the input stream searching for a JOB command.
;  571          0676  2 !
;  572          0677  2 UNTIL .CURRENT_COMMAND EQL K_JOB DO
;  573          0678  3     BEGIN
;  574          0679  3     IF NOT GET_RECORD() THEN RETURN TRUE;
;  575          0680  3     CURRENT_COMMAND = IDENTIFY_COMMAND_VERB(FALSE, LINE_DESC);
;  576          0681  2     END;
;  577          0682  2
;  578          0683  2
;  579          0684  2 ! Save the JOB command for error messages.
;  580          0685  2 !
;  581          0686  2 JOB_LENGTH = .INPUT_RAB[RAB$W_RSZ];
;  582          0687  2 CH$MOVE(.JOB_LENGTH, .INPUT_RAB[RAB$L_RBF], JOB_BUFFER);
;  583          0688  2
;  584          0689  2
;  585          0690  2 ! Parse the JOB command.
;  586          0691  2 !
;  587          0692  2 CURRENT_COMMAND = K_NONE;
;  588          0693  2 CLI$DCL_PARSE(LINE_DESC, INPSMBCLD, 0, GET_LINE_CONTINUATION);
;  589          0694  2
;  590          0695  2
;  591          0696  2 ! Free dynamic strings to ensure that jobs do not interfere with one another.
;  592          0697  2 !
;  593          0698  2 LIB$SFREE1_DD(VALUE_DESC);
;  594          0699  2 LIB$SFREE1_DD(LOG_FILE_DESC);
;  595          0700  2 LIB$SFREE1_DD(NAME_DESC);
;  596          0701  2 LIB$SFREE1_DD(USERNAME_DESC);
;  597          0702  2 LIB$SFREE1_DD(PASSWORD_DESC);
;  598          0703  2
;  599          0704  2
;  600          0705  2 ! Get the parameter, which is the username.
;  601          0706  2 !
;  602          0707  2 CLI$GET_VALUE(D_P1, USERNAME_DESC);
;  603          0708  2 IF .USERNAME_DESC[DSC$W_LENGTH] GTRU 12
;  604          0709  2 THEN
;  605          0710  2     SIGNAL(INPSMB$_INVUSER, 1, USERNAME_DESC);
;  606          0711  2
;  607          0712  2
;  608          0713  2 ! Get the /QUEUE qualifier.
;  609          0714  2 !
;  610          0715  2 PARSE_CALL(QUEUE, D_QUEUE, SJC$_QUEUE, $DESCRIPTOR('SYS$BATCH'));
;  611          0716  2 Q_MESSAGE = INPSMB$_FACILITY^16 + SHR$_INVQUAVAL + STS$K_SEVERE;
;  612          0717  2
;  613          0718  2
;  614          0719  2 ! Get the /AFTER qualifier.
;  615          0720  2 !
;  616          0721  2 PARSE_CALL(AFTER, D_AFTER);
;  617          0722  2
;  618          0723  2
;  619          0724  2 ! Get the /CHARACTERISTICS qualifier.
;  620          0725  2 !
;  621          0726  2 PARSE_CALL(CHARACTERISTICS, D_CHARACTERISTICS);
;  622          0727  2
;  623          0728  2
;  624          0729  2 ! Get the /CLI qualifier.
```

```
;  625     0730  2 !
;  626     0731  2 PARSE_CALL(FILENAME, D_CLI, SJC$_CLI, SJC$_NO_CLI);
;  627     0732  2
;  628     0733  2
;  629     0734  2 ! Get the /CPUTIME qualifier.
;  630     0735  2 !
;  631     0736  2 PARSE_CALL(CPUTIME, D_CPUTIME, SJC$_CPU_LIMIT, SJC$_NO_CPU_LIMIT);
;  632     0737  2
;  633     0738  2
;  634     0739  2 ! Get the /DELETE qualifier.
;  635     0740  2 !
;  636     0741  2 PARSE_CALL(IF_TRUE, D_DELETE, SJC$_DELETE_FILE);
;  637     0742  2
;  638     0743  2
;  639     0744  2 ! Get the /HOLD qualifier.
;  640     0745  2 !
;  641     0746  2 PARSE_CALL(IF_TRUE, D_HOLD, SJC$_HOLD);
;  642     0747  2
;  643     0748  2
;  644     0749  2 ! Get the /KEEP qualifier.
;  645     0750  2 !
;  646     0751  2 PARSE_CALL(IF_TRUE, D_KEEP, SJC$_NO_LOG_DELETE);
;  647     0752  2
;  648     0753  2
;  649     0754  2 ! Get the /LOG_FILE qualifier.
;  650     0755  2 !
;  651     0756  2 Q_VALUE_DESC = LOG_FILE_DESC;
;  652     0757  2 FLAGS[V_NO_LOG_FILE] = PARSE_CALL_VALUE(LOG_FILE, D_LOG_FILE);
;  653     0758  2
;  654     0759  2
;  655     0760  2 ! Get the /NAME qualifier.
;  656     0761  2 !
;  657     0762  2 Q_VALUE_DESC = NAME_DESC;
;  658     0763  2 PARSE_CALL(NAME, D_NAME);
;  659     0764  2 Q_VALUE_DESC = VALUE_DESC;
;  660     0765  2
;  661     0766  2
;  662     0767  2 ! Get the /NOTIFY qualifier.
;  663     0768  2 !
;  664     0769  2 PARSE_CALL(IF_TRUE, D_NOTIFY, SJC$_NOTIFY);
;  665     0770  2
;  666     0771  2
;  667     0772  2 ! Get the /PARAMETERS qualifier.
;  668     0773  2 !
;  669     0774  2 PARSE_CALL(PARAMETERS, D_PARAMETERS);
;  670     0775  2
;  671     0776  2
;  672     0777  2 ! Get the /PRINTER qualifier.
;  673     0778  2 !
;  674     0779  2 PARSE_CALL(PRINTER, D_PRINTER);
;  675     0780  2
;  676     0781  2
;  677     0782  2 ! Get the /PRIORITY qualifier.
;  678     0783  2 !
;  679     0784  2 PARSE_CALL(PRIORITY, D_PRIORITY);
;  680     0785  2
;  681     0786  2
```

```
;  682     0787  2 ! Get the /RESTART qualifier.
;  683     0788  2 !
;  684     0789  2 PARSE_CALL(IF_TRUE, D_RESTART, SJC$_RESTART);
;  685     0790  2
;  686     0791  2
;  687     0792  2 ! Get the /TRAILING_BLANKS qualifier.
;  688     0793  2 !
;  689     0794  2 FLAGS[V_TRAILING_BLANKS] = CLI$PRESENT(D_TRAILING_BLANKS);
;  690     0795  2
;  691     0796  2
;  692     0797  2 ! Get the /WSDEFAULT qualifier.
;  693     0798  2 !
;  694     0799  2 PARSE_CALL(WORKING_SET, D_WSDEFAULT, SJC$_WSDEFAULT, SJC$_NO_WSDEFAULT);
;  695     0800  2
;  696     0801  2
;  697     0802  2 ! Get the /WSEXTENT qualifier.
;  698     0803  2 !
;  699     0804  2 PARSE_CALL(WORKING_SET, D_WSEXTENT, SJC$_WSEXTENT, SJC$_NO_WSEXTENT);
;  700     0805  2
;  701     0806  2
;  702     0807  2 ! Get the /WSQUOTA qualifier.
;  703     0808  2 !
;  704     0809  2 PARSE_CALL(WORKING_SET, D_WSQUOTA, SJC$_WSQUOTA, SJC$_NO_WSQUOTA);
;  705     0810  2
;  706     0811  2
;  707     0812  2 ! Read the input stream for a PASSWORD command.
;  708     0813  2 !
;  709     0814  2 IF NOT GET_RECORD() THEN RETURN TRUE;
;  710     0815  2 CURRENT_COMMAND = IDENTIFY_COMMAND_VERB(TRUE, LINE_DESC);
;  711     0816  2 IF .CURRENT_COMMAND NEQ K_PASSWORD THEN SIGNAL(INPSMB$_MISSPASS);
;  712     0817  2
;  713     0818  2
;  714     0819  2 ! Parse the PASSWORD command.
;  715     0820  2 !
;  716     0821  2 CLI$DCL_PARSE(LINE_DESC, INPSMBCLD, 0, GET_LINE_CONTINUATION);
;  717     0822  2
;  718     0823  2
;  719     0824  2 ! Get the parameter, which is the password.
;  720     0825  2 !
;  721     0826  2 CLI$GET_VALUE(D_P1, PASSWORD_DESC);
;  722     0827  2 IF .PASSWORD_DESC[DSC$W_LENGTH] GTRU 31
;  723     0828  2 THEN
;  724     0829  2     SIGNAL(INPSMB$_INVPASS, 1, PASSWORD_DESC);
;  725     0830  2
;  726     0831  2
;  727     0832  2 ! Validate access to the specified username and password.
;  728     0833  2 !
;  729     0834  2 UAF_DESC[0] = %ALLOCATION(UAF_BUFFER);
;  730     0835  2 UAF_DESC[1] = UAF_BUFFER;
;  731     0836  2 STATUS_1 = LGI$VALIDATE(USERNAME_DESC, PASSWORD_DESC, UAF_DESC);
;  732     0837  2 IF NOT .STATUS_1
;  733     0838  2 THEN
;  734     0839  2     IF .STATUS_1 GEQ 0
;  735     0840  2         THEN SIGNAL(INPSMB$_OPENUAF, 0, .STATUS_1)
;  736     0841  2         ELSE SIGNAL(INPSMB$_USERVAL);
;  737     0842  2
;  738     0843  2
```

```
;  739          0844  2 IF NOT .FLAGS[V_NO_LOG_FILE]
;  740          0845  2 THEN
;  741          0846  2     BEGIN
;  742          0847  2
;  743          0848  2     ! Compute the log file default name string.
;  744          0849  2     !
;  745          0850  2     DNA_DESC[0] = %ALLOCATION(DNA_BUFFER);
;  746          0851  2     DNA_DESC[1] = DNA_BUFFER;
;  747        P 0852  2     $FAO(
;  748        P 0853  2         %DESCRIPTOR('!AC!AC.LOG'),
;  749        P 0854  2         DNA_DESC,
;  750        P 0855  2         DNA_DESC,
;  751        P 0856  2         UAF_BUFFER[UAF$T_DEFDEV],
;  752          0857  2         UAF_BUFFER[UAF$T_DEFDIR]);
;  753          0858  2
;  754          0859  2
;  755          0860  2     ! Compute the log file specification.
;  756          0861  2     !
;  757        P 0862  2     PARSE_CALL(NAME_AND_LOG_FILE,
;  758        P 0863  2         NAME_DESC, LOG_FILE_DESC,
;  759          0864  2         DNA_DESC, INPSMB$_INVLOGFIL);
;  760          0865  2     END;
;  761          0866  2
;  762          0867  2
;  763          0868  2 ! Compute the command file default name string.
;  764          0869  2 !
;  765          0870  2 DNA_DESC[0] = %ALLOCATION(DNA_BUFFER);
;  766          0871  2 DNA_DESC[1] = DNA_BUFFER;
;  767        P 0872  2 $FAO(
;  768        P 0873  2     %DESCRIPTOR('!AC!ACINPBATCH.COM'),
;  769        P 0874  2     DNA_DESC,
;  770        P 0875  2     DNA_DESC,
;  771        P 0876  2     UAF_BUFFER[UAF$T_DEFDEV],
;  772          0877  2     UAF_BUFFER[UAF$T_DEFDIR]);
;  773          0878  2
;  774          0879  2
;  775          0880  2 ! Create the output command file.
;  776          0881  2 !
;  777        P 0882  2 $FAB_INIT(FAB=OUTPUT_FAB,
;  778        P 0883  2     DNA=.DNA_DESC[1],
;  779        P 0884  2     DNS=.DNA_DESC[0],
;  780        P 0885  2     FAC=PUT,
;  781        P 0886  2     FNA=.NAME_DESC[DSC$A_POINTER],
;  782        P 0887  2     FNS=.NAME_DESC[DSC$W_LENGTH],
;  783        P 0888  2     FOP=SQO,
;  784        P 0889  2     NAM=OUTPUT_NAM,
;  785        P 0890  2     ORG=SEQ,
;  786        P 0891  2     RAT=CR,
;  787        P 0892  2     RFM=VAR,
;  788          0893  2     XAB=OUTPUT_XAB);
;  789        P 0894  2 $RAB_INIT(RAB=OUTPUT_RAB,
;  790        P 0895  2     FAB=OUTPUT_FAB,
;  791          0896  2     ROP=WBH);
;  792        P 0897  2 $NAM_INIT(NAM=OUTPUT_NAM,
;  793        P 0898  2     ESA=OUTPUT_RSA,
;  794        P 0899  2     ESS=NAM$C_MAXRSS,
;  795        P 0900  2     RSA=OUTPUT_RSA,
```

```
;  796          0901  2      RSS=NAM$C_MAXRSS);
;  797        P 0902  2  $XABPRO_INIT(XAB=OUTPUT_XAB,
;  798          0903  2      PRO=<RWED,RWED,,>);
;  799          0904  2  OUTPUT_XAB[XAB$L_UIC] = .UAF_BUFFER[UAF$L_UIC];
;  800          0905  2
;  801          0906  2
;  802          0907  3  IF NOT $CREATE(FAB=OUTPUT_FAB)
;  803          0908  2  THEN
;  804          0909  2      FILE_ERROR(
;  805          0910  2          INPSMB$_FACILITY^16 + SHR$_OPENOUT + STS$K_ERROR,
;  806          0911  2          OUTPUT_FAB,
;  807          0912  2          .OUTPUT_FAB[FAB$L_STS], .OUTPUT_FAB[FAB$L_STV]);
;  808          0913  2
;  809          0914  2
;  810          0915  3  IF NOT $CONNECT(RAB=OUTPUT_RAB)
;  811          0916  2  THEN
;  812          0917  2      FILE_ERROR(
;  813          0918  2          INPSMB$_FACILITY^16 + SHR$_OPENOUT + STS$K_ERROR,
;  814          0919  2          OUTPUT_FAB,
;  815          0920  2          .OUTPUT_RAB[RAB$L_STS], .OUTPUT_RAB[RAB$L_STV]);
;  816          0921  2
;  817          0922  2
;  818          0923  2  ! Read the input stream into the command file until a JOB or EOJ command.
;  819          0924  2  !
;  820          0925  2  WHILE TRUE DO
;  821          0926  3      BEGIN
;  822          0927  3      LOCAL
;  823          0928  3          RECORD_LENGTH;                  ! Input record length
;  824          0929  3
;  825          0930  3
;  826          0931  3      ! Get the next record.  If it is JOB or EOJ, we are finished.
;  827          0932  3      !
;  828          0933  3      IF NOT GET_RECORD() THEN EXITLOOP;
;  829          0934  3      CURRENT_COMMAND = IDENTIFY_COMMAND_VERB(FALSE, LINE_DESC);
;  830          0935  3      IF .CURRENT_COMMAND EQL K_JOB OR .CURRENT_COMMAND EQL K_EOJ THEN EXITLOOP;
;  831          0936  3
;  832          0937  3
;  833          0938  3      ! Trim trailing blanks if requested.
;  834          0939  3      !
;  835          0940  3      RECORD_LENGTH = .INPUT_RAB[RAB$W_RSZ];
;  836          0941  3      IF NOT .FLAGS[V_TRAILING_BLANKS]
;  837          0942  3      THEN
;  838          0943  4          BEGIN
;  839          0944  4          WHILE .RECORD_LENGTH GTR 0 DO
;  840          0945  5              BEGIN
;  841          0946  5              IF CH$RCHAR(.INPUT_RAB[RAB$L_RBF] + .RECORD_LENGTH - 1) NEQ %C' '
;  842          0947  5              THEN EXITLOOP;
;  843          0948  5              RECORD_LENGTH = .RECORD_LENGTH - 1;
;  844          0949  4              END;
;  845          0950  3          END;
;  846          0951  3
;  847          0952  3
;  848          0953  3      ! Copy the record to the output command file.
;  849          0954  3      !
;  850          0955  3      OUTPUT_RAB[RAB$W_RSZ] = .RECORD_LENGTH;
;  851          0956  3      OUTPUT_RAB[RAB$L_RBF] = .INPUT_RAB[RAB$L_RBF];
;  852          0957  4      IF NOT $PUT(RAB=OUTPUT_RAB)
```

```
;  853     0958  3      THEN
;  854     0959  3          FILE_ERROR(
;  855     0960  3              INPSMB$_FACILITY^16 + SHR$_WRITEERR + STS$K_ERROR,
;  856     0961  3              OUTPUT_FAB,
;  857     0962  3              .OUTPUT_RAB[RAB$L_STS], .OUTPUT_RAB[RAB$L_STV]);
;  858     0963  3      END;
;  859     0964  3
;  860     0965  3
;  861     0966  2  ! Close the output command file.
;  862     0967  2  !
;  863     0968  2  IF NOT $CLOSE(FAB=OUTPUT_FAB)
;  864     0969  2  THEN
;  865     0970  2      FILE_ERROR(
;  866     0971  2          INPSMB$_FACILITY^16 + SHR$_CLOSEOUT + STS$K_ERROR,
;  867     0972  2          OUTPUT_FAB,
;  868     0973  2          .OUTPUT_FAB[FAB$L_STS], .OUTPUT_FAB[FAB$L_STV]);
;  869     0974  2
;  870     0975  2
;  871     0976  2  ! Set up the user identification item.
;  872     0977  2  !
;  873     0978  2  Q_DCURSOR[0,0,32,0] = .UAF_BUFFER[UAF$L_UIC];
;  874     0979  2  CH$MOVE(
;  875     0980  2      UAF$S_USERNAME,
;  876     0981  2      UAF_BUFFER[UAF$T_USERNAME],
;  877     0982  2      Q_DCURSOR[4,0,0,0]);
;  878     0983  2  CH$MOVE(
;  879     0984  2      UAF$S_ACCOUNT,
;  880     0985  2      UAF_BUFFER[UAF$T_ACCOUNT],
;  881     0986  2      Q_DCURSOR[16,0,0,0]);
;  882     0987  2  Q_DCURSOR[24,0,8,0] = .UAF_BUFFER[UAF$B_PRI];
;  883     0988  2
;  884     0989  2
;  885     0990  2  ! Add the remaining items and finish the list.
;  886     0991  2  !
;  887     0992  2  Q_ICURSOR[0,0,16,0] = NAM$S_DVI + FID$C_LENGTH + FID$C_LENGTH;
;  888     0993  2  Q_ICURSOR[2,0,16,0] = SJC$_FILE_IDENTIFICATION;
;  889     0994  2  Q_ICURSOR[4,0,32,0] = OUTPUT_NAM[NAM$T_DVI];
;  890     0995  2  Q_ICURSOR[8,0,32,0] = 0;
;  891     0996  2
;  892     0997  2  Q_ICURSOR[12,0,16,0] = 25;
;  893     0998  2  Q_ICURSOR[14,0,16,0] = SJC$_USER_IDENTIFICATION;
;  894     0999  2  Q_ICURSOR[16,0,32,0] = .Q_DCURSOR;
;  895     1000  2  Q_ICURSOR[20,0,32,0] = 0;
;  896     1001  2
;  897     1002  2  Q_ICURSOR[24,0,32,0] = 0;
;  898     1003  2
;  899     1004  2
;  900     1005  2  ! Submit the output command file.
;  901     1006  2  !
;  902   P 1007  2  STATUS_2 = $SNDJBCW(
;  903   P 1008  2      FUNC=SJC$_ENTER_FILE,
;  904   P 1009  2      IOSB=IOSB,
;  905     1010  2      ITMLST=ITEM_BUFFER);
;  906     1011  2  IF .STATUS_2 THEN STATUS_2 = .IOSB;
;  907     1012  2  IF NOT .STATUS_2
;  908     1013  2  THEN
;  909     1014  2      SIGNAL(INPSMB$_ENTFIL, 0, .STATUS_2);
```

```
;  910          1015  2
;  911          1016  2
;  912          1017  2 ! Terminate if this was end of file.
;  913          1018  2 !
;  914          1019  2 IF NOT .INPUT_RAB[RAB$L_STS] THEN RETURN TRUE;
;  916          1020  2 FALSE
;  916          1021  1 END;


                                                    003CC P.ABT:   .ASCII  \SYS$BATCH\                          ;
                                                    003D5          .BLKB   3
                                          00000009  003D8 P.ABS:   .LONG   9                                    ;
                                          00000000' 003DC          .ADDRESS P.ABT                              ;
                                47 4F 4C 2E 43 41 21 43 41 21     003E0 P.ABV:   .ASCII  \!AC!AC.LOG\                        ;
                                                    003FA          .BLKB   2
                                          0000000A  003FC P.ABU:   .LONG   10                                   ;
                                          00000000' 00400          .ADDRESS P.ABV                              ;
2E 48 43 54 41 42 50 4E 49 43 41 21 43 41 21      003F4 P.ABX:   .ASCII  \!AC!ACINPBATCH.COM\                  ;
                                             4D 4F 43 00403                                                   ;
                                                    00406          .BLKB   2
                                          00000012  00408 P.ABW:   .LONG   18                                   ;
                                          00000000' 0040C          .ADDRESS P.ABX                              ;

                                                                   $RMS_PTR=          OUTPUT_FAB
                                                                   $RMS_PTR=          OUTPUT_RAB
                                                                   $RMS_PTR=          OUTPUT_NAM
                                                                   $RMS_PTR=          OUTPUT_XAB
                                                                   .EXTRN  PARSE_QUEUE, PARSE_AFTER
                                                                   .EXTRN  PARSE_CHARACTERISTICS
                                                                   .EXTRN  PARSE_FILENAME, PARSE_CPUTIME
                                                                   .EXTRN  PARSE_IF_TRUE, PARSE_LOG_FILE
                                                                   .EXTRN  PARSE_NAME, PARSE_PARAMETERS
                                                                   .EXTRN  PARSE_PRINTER, PARSE_PRIORITY
                                                                   .EXTRN  PARSE_WORKING_SET
                                                                   .EXTRN  SYS$FAO, PARSE_NAME_AND_LOG_FILE
                                                                   .EXTRN  SYS$CREATE, SYS$PUT
                                                                   .EXTRN  SYS$SNDJBCW

                                                    0FFC 00000 PROCESSING_LOOP:
                                                               .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11     ; 0616
                                57      FBF6    CF  9E 00002       MOVAB   D_P1, R7                             ;
                                56      0000'   CF  9E 00007       MOVAB   $RMS_PTR, R6                         ;
                                5E      E95C    CE  9E 0000C       MOVAB   -5798(SP), SP                        ;
                                6D      FF54    CF  9E 00011       MOVAB   PROCESSING_LOOP_HANDLER, (FP)        ; 0663
                                5B      F800    CD  9E 00016       MOVAB   ITEM_BUFFER, Q_ICURSOR               ; 0672
                                5A      06A4    CE  9E 0001B       MOVAB   DATA_BUFFER, Q_DCURSOR               ;
                                59 00000000*    8F  D0 00020       MOVL    #<<<INPSMB$_FACILITY@16>!4488>!4>, -  ;
                                                                           Q_MESSAGE                            ;
                                58      02BC    C6  9E 00027       MOVAB   VALUE_DESC, Q_VALUE_DESC             ;
                                01      02E4    C6  D1 0002C 1$:   CMPL    CURRENT_COMMAND, #1                  ; 0677
                                                1B  13 00031       BEQL    3$                                   ;
                          0000V CF              00  FB 00033       CALLS   #0, GET_RECORD                       ; 0679
                                03              50  E8 00038       BLBS    R0, 2$                               ;
                                            047A 31  00038         BRW     20$                                  ;
                                              5E DD 0003E 2$:      PUSHL   SP                                   ; 0680
                                              7E D4 00040          CLRL    -(SP)                                ;
```

```
                    0000V CF          02 FB 00042          CALLS   #2, IDENTIFY_COMMAND_VERB                        ;
                    02E4  C6          50 D0 00047          MOVL    R0, CURRENT_COMMAND                             ;
                                      DE 11 0004C          BRB     1$                                              ; 0677
                    024C  C6    FDDE  C6 3C 0004E 3$:      MOVZWL  INPUT_RAB+34, JOB_LENGTH                        ; 0686
              0250  C6    FDE4  D6    024C  C6 28 00055    MOVC3   JOB_LENGTH, @INPUT_RAB+40, JOB_BUFFER           ; 0687
                                      02E4  C6 D4 0005F    CLRL    CURRENT_COMMAND                                 ; 0692
                                0000V CF 9F 00063          PUSHAB  GET_LINE_CONTINUATION                           ; 0693
                                      7E D4 00067          CLRL    -(SP)                                           ;
                                0000G CF 9F 00069          PUSHAB  INPSMBCLD                                       ;
                                   0C AE 9F 0006D          PUSHAB  LINE_DESC                                       ;
                 00000000G  00        04 FB 00070          CALLS   #4, CLI$DCL_PARSE                               ;
                                02BC  C6 9F 00077          PUSHAB  VALUE_DESC                                      ; 0698
                 00000000G  00        01 FB 0007B          CALLS   #1, LIB$SFREE1_DD                               ;
                                02C4  C6 9F 00082          PUSHAB  LOG_FILE_DESC                                   ; 0699
                 00000000G  00        01 FB 00086          CALLS   #1, LIB$SFREE1_DD                               ;
                                02CC  C6 9F 0008D          PUSHAB  NAME_DESC                                       ; 0700
                 00000000G  00        01 FB 00091          CALLS   #1, LIB$SFREE1_DD                               ;
                                02D4  C6 9F 00098          PUSHAB  USERNAME_DESC                                   ; 0701
                 00000000G  00        01 FB 0009C          CALLS   #1, LIB$SFREE1_DD                               ;
                                02DC  C6 9F 000A3          PUSHAB  PASSWORD_DESC                                   ; 0702
                 00000000G  00        01 FB 000A7          CALLS   #1, LIB$SFREE1_DD                               ;
                                02D4  C6 9F 000AE          PUSHAB  USERNAME_DESC                                   ; 0707
                                      57 DD 000B2          PUSHL   R7                                              ;
                 00000000G  00        02 FB 000B4          CALLS   #2, CLI$GET_VALUE                               ;
                              0C  02D4  C6 B1 000BB          CMPW    USERNAME_DESC, #12                              ; 0708
                                      13 1B 000C0          BLEQU   4$                                              ;
                                02D4  C6 9F 000C2          PUSHAB  USERNAME_DESC                                   ; 0710
                                      01 DD 000C6          PUSHL   #1                                              ;
                          00000000G   8F DD 000C8          PUSHL   #INPSMB$_INVUSER                                ;
          00000000G  00               03 FB 000CE          CALLS   #3, LIB$SIGNAL                                  ;
                           03CC  C7 9F 000D5 4$:           PUSHAB  P.ABS                                           ; 0715
                        7E   86  8F 9A 000D9               MOVZBL  #134, -(SP)                                     ;
                           00DC  C7 9F 000DD               PUSHAB  D_QUEUE                                         ;
                     0000G CF    03 FB 000E1               CALLS   #3, PARSE_QUEUE                                 ;
                           59 00000000*  8F DD 000E6        MOVL    #<<<INPSMB$_FACILITY@16>+4904>+4>, -            ; 0716
                                                                    Q_MESSAGE                                       ;
                              10  A7 9F 000ED              PUSHAB  D_AFTER                                         ; 0721
                     0000G CF     01 FB 000F0              CALLS   #1, PARSE_AFTER                                 ;
                              28  A7 9F 000F5              PUSHAB  D_CHARACTERISTICS                               ; 0726
                     0000G CF     01 FB 000F8              CALLS   #1, PARSE_CHARACTERISTICS                       ;
                                  12 DD 000FD              PUSHL   #18                                             ; 0731
                                  11 DD 000FF              PUSHL   #17                                             ;
                              34  A7 9F 00101              PUSHAB  D_CLI                                           ;
                     0000G CF     03 FB 00104              CALLS   #3, PARSE_FILENAME                              ;
                                  16 DD 00109              PUSHL   #22                                             ; 0736
                                  15 DD 0010B              PUSHL   #21                                             ;
                              44  A7 9F 0010D              PUSHAB  D_CPUTIME                                       ;
                     0000G CF     03 FB 00110              CALLS   #3, PARSE_CPUTIME                               ;
                                  18 DD 00115              PUSHL   #24                                             ; 0741
                              54  A7 9F 00117              PUSHAB  D_DELETE                                        ;
                     0000G CF     02 FB 0011A              CALLS   #2, PARSE_IF_TRUE                               ;
                           7E     47 8F 9A 0011F           MOVZBL  #71, -(SP)                                      ; 0746
                              60  A7 9F 00123              PUSHAB  D_HOLD                                          ;
                     0000G CF     02 FB 00126              CALLS   #2, PARSE_IF_TRUE                               ;
                           7E     60 8F 9A 0012B           MOVZBL  #96, -(SP)                                      ; 0751
                              6C  A7 9F 0012F              PUSHAB  D_KEEP                                          ;
                     0000G CF     02 FB 00132              CALLS   #2, PARSE_IF_TRUE                               ;
```

```
                                58      02C4   C6  9E 00137         MOVAB   LOG_FILE_DESC, Q_VALUE_DESC                ; 0756
                                   7C   A7  9F 0013C               PUSHAB  D_LOG_FILE                                ; 0757
                         0000G  CF           01  FB 0013F         CALLS   #1, PARSE_LOG_FILE
    02A4   C6            01             00  50  F0 00144         INSV    R0, #0, #1, FLAGS
                                58      02CC   C6  9E 0014B         MOVAB   NAME_DESC, Q_VALUE_DESC                   ; 0762
                                   0088 C7  9F 00150               PUSHAB  D_NAME                                    ; 0763
                         0000G  CF           01  FB 00154         CALLS   #1, PARSE_NAME
                                58      02BC   C6  9E 00159         MOVAB   VALUE_DESC, Q_VALUE_DESC                  ; 0764
                                7E         6C   8F  9A 0015E         MOVZBL  #108, -(SP)                               ; 0769
                                   0098 C7  9F 00162               PUSHAB  D_NOTIFY
                         0000G  CF           02  FB 00166         CALLS   #2, PARSE_IF_TRUE
                                   00AC C7  9F 0016B               PUSHAB  D_PARAMETERS                              ; 0774
                         0000G  CF           01  FB 0016F         CALLS   #1, PARSE_PARAMETERS
                                   00BC C7  9F 00174               PUSHAB  D_PRINTER                                 ; 0779
                         0000G  CF           01  FB 00178         CALLS   #1, PARSE_PRINTER
                                   00CC C7  9F 0017D               PUSHAB  D_PRIORITY                                ; 0784
                         0000G  CF           01  FB 00181         CALLS   #1, PARSE_PRIORITY
                                7E         8A   8F  9A 00186         MOVZBL  #138, -(SP)                               ; 0789
                                   00EC C7  9F 0018A               PUSHAB  D_RESTART
                         0000G  CF           02  FB 0018E         CALLS   #2, PARSE_IF_TRUE
                                   0104 C7  9F 00193               PUSHAB  D_TRAILING_BLANKS                         ; 0794
                000000000G  00             01  FB 00197         CALLS   #1, CLI$PRESENT
    02A4   C6            01             02  50  F0 0019E         INSV    R0, #2, #1, FLAGS
                                7E         98   8F  9A 001A5         MOVZBL  #152, -(SP)                               ; 0799
                                7E         97   8F  9A 001A9         MOVZBL  #151, -(SP)
                                   0118 C7  9F 001AD               PUSHAB  D_WSDEFAULT
                         0000G  CF           03  FB 001B1         CALLS   #3, PARSE_WORKING_SET
                                7E         9A   8F  9A 001B6         MOVZBL  #154, -(SP)                               ; 0804
                                7E         99   8F  9A 001BA         MOVZBL  #153, -(SP)
                                   0128 C7  9F 001BE               PUSHAB  D_WSEXTENT
                         0000G  CF           03  FB 001C2         CALLS   #3, PARSE_WORKING_SET
                                7E         9C   8F  9A 001C7         MOVZBL  #156, -(SP)                               ; 0809
                                7E         9B   8F  9A 001CB         MOVZBL  #155, -(SP)
                                   0138 C7  9F 001CF               PUSHAB  D_WSQUOTA
                         0000G  CF           03  FB 001D3         CALLS   #3, PARSE_WORKING_SET
                         0000V  CF           00  FB 001D8         CALLS   #0, GET_RECORD                            ; 0814
                                03             50  E8 001DD         BLBS    R0, 5$
                                      02D5     31  001E0         BRW     20$
                                         5E    DD  001E3  5$:     PUSHL   SP                                        ; 0815
                                         01    DD  001E5         PUSHL   #1
                         0000V  CF           02  FB 001E7         CALLS   #2, IDENTIFY_COMMAND_VERB
                         02E4   C6           50  D0 001EC         MOVL    R0, CURRENT_COMMAND
                                05       02E4   C6  D1 001F1         CMPL    CURRENT_COMMAND, #5                       ; 0816
                                         0D     13  001F6         BEQL    6$
                                00000000G  8F   DD  001F8         PUSHL   #INPSMB$_MISSPASS
                00000000G  00            01  FB 001FE         CALLS   #1, LIB$SIGNAL
                                   0000V CF  9F 00205  6$:     PUSHAB  GET_LINE_CONTINUATION                     ; 0821
                                         7E    D4  00209         CLRL    -(SP)
                                   0000G CF  9F 0020B         PUSHAB  INPSMBCLD
                                      0C   AE  9F 0020F         PUSHAB  LINE_DESC
                00000000G  00            04  FB 00212         CALLS   #4, CLI$DCL_PARSE
                                   02DC  C6  9F 00219         PUSHAB  PASSWORD_DESC                             ; 0826
                                         57    DD  0021D         PUSHL   R7
                00000000G  00            02  FB 0021F         CALLS   #2, CLI$GET_VALUE
                                1F       02DC   C6  B1 00226         CMPW    PASSWORD_DESC, #31                        ; 0827
                                         13     1B  0022B         BLEQU   7$
                                   02DC  C6  9F 0022D         PUSHAB  PASSWORD_DESC                             ; 0829
```

```
                                         01  DD 00231           PUSHL   #1                                    ;
                               00000000G 8F  DD 00233           PUSHL   #INPSMB$_INVPASS                      ;
              00000000G  00              03  FB 00239           CALLS   #3, LIB$SIGNAL                        ;
                   0118  CE         0584 8F  3C 00240 7$:       MOVZWL  #1412, UAF_DESC                       ; 0834
                   011C  CE         0120 CE  9E 00247           MOVAB   UAF_BUFFER, UAF_DESC+4                ; 0835
                                    0118 CE  9F 0024E           PUSHAB  UAF_DESC                              ; 0836
                                    02DC C6  9F 00252           PUSHAB  PASSWORD_DESC                         ;
                                    02D4 C6  9F 00256           PUSHAB  USERNAME_DESC                         ;
              00000000G  00              03  FB 0025A           CALLS   #3, LGI$VALIDATE                      ;
                         24              50  E8 00261           BLBS    STATUS_1, 9$                          ; 0837
                                         50  D5 00264           TSTL    STATUS_1                              ; 0839
                                         13  19 00266           BLSS    8$                                    ;
                                         50  DD 00268           PUSHL   STATUS_1                              ; 0840
                                         7E  D4 0026A           CLRL    -(SP)                                 ;
                               00000000G 8F  DD 0026C           PUSHL   #INPSMB$_OPENUAF                      ;
              00000000G  00              03  FB 00272           CALLS   #3, LIB$SIGNAL                        ;
                                         0D  11 00279           BRB     9$                                    ;
                               00000000G 8F  DD 0027B 8$:       PUSHL   #INPSMB$_USERVAL                      ; 0841
              00000000G  00              01  FB 00281           CALLS   #1, LIB$SIGNAL                        ;
                         39         02A4 C6  E8 00288 9$:       BLBS    FLAGS, 10$                            ; 0844
                     10  AE           FF 8F  9A 0028D           MOVZBL  #255, DNA_DESC                        ; 0850
                     14  AE           18 AE  9E 00292           MOVAB   DNA_BUFFER, DNA_DESC+4                ; 0851
                                    01B4 CE  9F 00297           PUSHAB  UAF_BUFFER+148                        ; 0857
                                    0198 CE  9F 0029B           PUSHAB  UAF_BUFFER+116                        ;
                                      18 AE  9F 0029F           PUSHAB  DNA_DESC                              ;
                                      1C AE  9F 002A2           PUSHAB  DNA_DESC                              ;
                                    03E0 C7  9F 002A5           PUSHAB  P.ABU                                 ;
              00000000G  00              05  FB 002A9           CALLS   #5, SYS$FAO                           ;
                               00000000G 8F  DD 002B0           PUSHL   #INPSMB$_INVLOGFIL                    ; 0864
                                      14 AE  9F 002B6           PUSHAB  DNA_DESC                              ;
                                    02C4 C6  9F 002B9           PUSHAB  LOG_FILE_DESC                         ;
                                    02CC C6  9F 002BD           PUSHAB  NAME_DESC                             ;
                  0000G  CF              04  FB 002C1           CALLS   #4, PARSE_NAME_AND_LOG_FILE           ;
                     10  AE           FF 8F  9A 002C6 10$:      MOVZBL  #255, DNA_DESC                        ; 0870
                     14  AE           18 AE  9E 002CB           MOVAB   DNA_BUFFER, DNA_DESC+4                ; 0871
                                    01B4 CE  9F 002D0           PUSHAB  UAF_BUFFER+148                        ; 0877
                                    0198 CE  9F 002D4           PUSHAB  UAF_BUFFER+116                        ;
                                      18 AE  9F 002D8           PUSHAB  DNA_DESC                              ;
                                      1C AE  9F 002DB           PUSHAB  DNA_DESC                              ;
                                    03FC C7  9F 002DE           PUSHAB  P.ABW                                 ;
              00000000G  00              05  FB 002E2           CALLS   #5, SYS$FAO                           ;
 0050  8F        00      6E              00  2C 002E9           MOVC5   #0, (SP), #0, #80, $RMS_PTR           ; 0893
                                         66     002F0                                                         ;
                         66         5003 8F  B0 002F1           MOVW    #20483, $RMS_PTR                      ;
                     04  A6           40 8F  9A 002F6           MOVZBL  #64, $RMS_PTR+4                       ;
                     16  A6              01  90 002FB           MOVB    #1, $RMS_PTR+22                       ;
                     1D  A6         0200 8F  B0 002FF           MOVW    #512, $RMS_PTR+29                     ;
                     1F  A6              02  90 00305           MOVB    #2, $RMS_PTR+31                       ;
                     24  A6         00F4 C6  9E 00309           MOVAB   OUTPUT_XAB, $RMS_PTR+36               ;
                     28  A6         0094 C6  9E 0030F           MOVAB   OUTPUT_NAM, $RMS_PTR+40               ;
                     2C  A6         02D0 C6  D0 00315           MOVL    NAME_DESC+4, $RMS_PTR+44              ;
                     30  A6           14 AE  D0 0031B           MOVL    DNA_DESC+4, $RMS_PTR+48               ;
                     34  A6         02CC C6  90 00320           MOVB    NAME_DESC, $RMS_PTR+52                ;
                     35  A6           10 AE  90 00326           MOVB    DNA_DESC, $RMS_PTR+53                 ;
 0044  8F        00      6E              00  2C 0032B           MOVC5   #0, (SP), #0, #68, $RMS_PTR           ; 0896
                                      50 A6     00332                                                         ;
                     50  A6         4401 8F  B0 00334           MOVW    #17409, $RMS_PTR                      ;
```

```
                          54  A6      0400  8F  3C 0033A            MOVZWL  #1024, $RMS_PTR+4                      ;
                        008C  C6            66  9E 00340            MOVAB   OUTPUT_FAB, $RMS_PTR+60                ;
  0060  8F          00        6E            00  2C 00345            MOVC5   #0, (SP), #0, #96, $RMS_PTR            ; 0901
                                      0094  C6     0034C                                                           ;
                        0094  C6      6002  8F  B0 0034F            MOVW    #24578, $RMS_PTR                       ;
                        0096  C6            01  8E 00356            MNEGB   #1, $RMS_PTR+2                         ;
                        0098  C6      014C  C6  9E 0035B            MOVAB   OUTPUT_RSA, $RMS_PTR+4                 ;
                        009E  C6            01  8E 00362            MNEGB   #1, $RMS_PTR+10                        ;
                        00A0  C6      014C  C6  9E 00367            MOVAB   OUTPUT_RSA, $RMS_PTR+12                ;
  0058  8F          00        6E            00  2C 0036E            MOVC5   #0, (SP), #0, #88, $RMS_PTR            ; 0903
                                      00F4  C6     00375                                                           ;
                        00F4  C6      5813  8F  B0 00378            MOVW    #22547, $RMS_PTR                       ;
                        00FC  C6      FF00  8F  B0 0037F            MOVW    #-256, $RMS_PTR+8                      ;
                        0100  C6      0144  CE  D0 00386            MOVL    UAF_BUFFER+36, OUTPUT_XAB+12           ; 0904
                                            56  DD 0038D            PUSHL   R6                                     ; 0907
                   00000000G  00            01  FB 0038F            CALLS   #1, SYS$CREATE                         ;
                              11            50  E8 00396            BLBS    R0, 11$                                ;
                              7E        08  A6  7D 00399            MOVQ    OUTPUT_FAB+8, -(SP)                    ; 0912
                                            56  DD 0039D            PUSHL   R6                                     ; 0909
                                 00000000*  8F  DD 0039F            PUSHL   #<<<INPSMB$_FACILITY@16>+4256>+2>      ; 0910
                       0000V  CF            04  FB 003A5            CALLS   #4, FILE_ERROR                         ;
                                        50  A6  9F 003AA  11$:      PUSHAB  OUTPUT_RAB                             ; 0915
                   00000000G  00            01  FB 003AD            CALLS   #1, SYS$CONNECT                        ;
                              11            50  E8 003B4            BLBS    R0, 13$                                ;
                              7E        58  A6  7D 003B7            MOVQ    OUTPUT_RAB+8, -(SP)                    ; 0920
                                            56  DD 003BB            PUSHL   R6                                     ; 0917
                                 00000000*  8F  DD 003BD            PUSHL   #<<<INPSMB$_FACILITY@16>+4256>+2>      ; 0918
                       0000V  CF            04  FB 003C3  12$:      CALLS   #4, FILE_ERROR                         ;
                       0000V  CF            00  FB 003C8  13$:      CALLS   #0, GET_RECORD                         ; 0933
                              5E            50  E9 003CD            BLBC    R0, 16$                                ;
                                            5E  DD 003D0            PUSHL   SP                                     ; 0934
                                            7E  D4 003D2            CLRL    -(SP)                                  ;
                       0000V  CF            02  FB 003D4            CALLS   #2, IDENTIFY_COMMAND_VERB              ;
                        02E4  C6            50  D0 003D9            MOVL    R0, CURRENT_COMMAND                    ;
                              01      02E4  C6  D1 003DE            CMPL    CURRENT_COMMAND, #1                    ; 0935
                                            49  13 003E3            BEQL    16$                                    ;
                              03      02E4  C6  D1 003E5            CMPL    CURRENT_COMMAND, #3                    ;
                                            42  13 003EA            BEQL    16$                                    ;
                              51      FDDE  C6  3C 003EC            MOVZWL  INPUT_RAB+34, RECORD_LENGTH            ; 0940
                  12    02A4  C6            02  E0 003F1            BBS     #2, FLAGS, 15$                         ; 0941
                                            10  15 003F7  14$:      BLEQ    15$                                    ; 0944
                  50          51      FDE4  C6  C1 003F9            ADDL3   INPUT_RAB+40, RECORD_LENGTH, R0        ; 0946
                              20        FF  A0  91 003FF            CMPB    -1(R0), #32                            ;
                                            04  12 00403            BNEQ    15$                                    ;
                                            51  D7 00405            DECL    RECORD_LENGTH                          ; 0948
                                            EE  11 00407            BRB     14$                                    ; 0944
                          72  A6            51  B0 00409  15$:      MOVW    RECORD_LENGTH, OUTPUT_RAB+34           ; 0955
                          78  A6      FDE4  C6  D0 0040D            MOVL    INPUT_RAB+40, OUTPUT_RAB+40            ; 0956
                                        50  A6  9F 00413            PUSHAB  OUTPUT_RAB                             ; 0957
                   00000000G  00            01  FB 00416            CALLS   #1, SYS$PUT                            ;
                              A8            50  E8 0041D            BLBS    R0, 13$                                ;
                              7E        58  A6  7D 00420            MOVQ    OUTPUT_RAB+8, -(SP)                    ; 0962
                                            56  DD 00424            PUSHL   R6                                     ; 0959
                                 00000000*  8F  DD 00426            PUSHL   #<<<INPSMB$_FACILITY@16>+4304>+2>      ; 0960
                                            95  11 0042C            BRB     12$                                    ;
                                            56  DD 0042E  16$:      PUSHL   R6                                     ; 0968
                   00000000G  00            01  FB 00430            CALLS   #1, SYS$CLOSE                          ;
```

```
                        11              50  E8  00437          BLBS    R0, 17$                                                 ;
                        7E        08    A6  7D  0043A          MOVQ    OUTPUT_FAB+8, -(SP)                                     ; 0973
                                        56  DD  0043E          PUSHL   R6                                                      ; 0970
                                  00000000* 8F  DD  00440          PUSHL   #<<<INPSMB$_FACILITY&16>+4184>+2>                       ; 0971
                    0000V       CF        04  FB  00446          CALLS   #4, FILE_ERROR
                        6A        0144      CE  D0  0044B 17$:     MOVL    UAF_BUFFER+36, (Q_DCURSOR)                              ; 0978
                04  AA    0124   CE        20  28  00450          MOVC3   #32, UAF_BUFFER+4, 4(Q_DCURSOR)                         ; 0982
                10  AA    0154   CE        20  28  00457          MOVC3   #32, UAF_BUFFER+52, 16(Q_DCURSOR)                       ; 0986
                          18  AA      0324  CE  90  0045E          MOVB    UAF_BUFFER+516, 24(Q_DCURSOR)                           ; 0987
                        6B  0027001C      8F  D0  00464          MOVL    #2555932, (Q_ICURSOR)                                   ; 0992
                      04    AB      00A8  C6  9E  0046B          MOVAB   OUTPUT_NAM+20, 4(Q_ICURSOR)                             ; 0994
                                        08  AB  D4  00471          CLRL    8(Q_ICURSOR)                                            ; 0995
                      0C    AB  00960019  8F  D0  00474          MOVL    #9830425, 12(Q_ICURSOR)                                 ; 0997
                      10    AB            5A  D0  0047C          MOVL    Q_DCURSOR, 16(Q_ICURSOR)                                ; 0999
                                      14  AB  7C  00480          CLRQ    20(Q_ICURSOR)                                           ; 1000
                                          7E  7C  00483          CLRQ    -(SP)                                                   ; 1010
                                      10  AE  9F  00485          PUSHAB  IOSB
                                    F800  CD  9F  00488          PUSHAB  ITEM_BUFFER
                        7E                13  7D  0048C          MOVQ    #19, -(SP)
                                          7E  D4  0048F          CLRL    -(SP)
                00000000G   00            07  FB  00491          CALLS   #7, SYS$SNDJBCW
                            07            50  E9  00498          BLBC    STATUS_2, 18$                                           ; 1011
                            50      08    AE  D0  0049B          MOVL    IOSB, STATUS_2
                            11            50  E8  0049F          BLBS    STATUS_2, 19$                                           ; 1012
                                          50  DD  004A2 18$:     PUSHL   STATUS_2                                                ; 1014
                                          7E  D4  004A4          CLRL    -(SP)
                                  00000000G 8F  DD  004A6          PUSHL   #INPSMB$_ENTFIL
                00000000G   00            03  FB  004AC          CALLS   #3, LIB$SIGNAL
                            04      FDC4  C6  E8  004B3 19$:     BLBS    INPUT_RAB+8, 21$                                        ; 1019
                            50            01  D0  004B8 20$:     MOVL    #1, R0                                                  ;
                                              04  004BB          RET                                                             ;
                                          50  D4  004BC 21$:     CLRL    R0                                                      ; 1021
                                              04  004BE          RET                                                             ;

; Routine Size:  1215 bytes,    Routine Base:  CODE + 0410
```

```
;  918     1022  1 ROUTINE GET_RECORD=
;  919     1023  1
;  920     1024  1 !++
;  921     1025  1 !
;  922     1026  1 ! FUNCTIONAL DESCRIPTION:
;  923     1027  1 !       This routine gets the next record from the input stream.
;  924     1028  1 !
;  925     1029  1 ! INPUT PARAMETERS:
;  926     1030  1 !       NONE
;  927     1031  1 !
;  928     1032  1 ! IMPLICIT INPUTS:
;  929     1033  1 !       NONE
;  930     1034  1 !
;  931     1035  1 ! OUTPUT PARAMETERS:
;  932     1036  1 !       NONE
;  933     1037  1 !
;  934     1038  1 ! IMPLICIT OUTPUTS:
;  935     1039  1 !       NONE
;  936     1040  1 !
;  937     1041  1 ! ROUTINE VALUE:
;  938     1042  1 !       Completion status.
;  939     1043  1 !
;  940     1044  1 ! SIDE EFFECTS:
;  941     1045  1 !       NONE
;  942     1046  1 !
;  943     1047  1 !--
;  944     1048  1
;  945     1049  2 BEGIN
;  946     1050  2 IF .CARD_CHANNEL NEQ 0
;  947     1051  2 THEN
;  948     1052  3     BEGIN
;  949     1053  3     LOCAL
;  950     1054  3         STATUS;
;  951     1055  3
;  952     1056  3
;  953     1057  3     IF .FLAGS[V_SECOND_BUFFER]
;  954     1058  3     THEN
;  955     1059  4         BEGIN
;  956     1060  4
;  957     1061  4         ! The second buffer had the pending read.  Wait for it to complete
;  958     1062  4         ! and examine the IOSB for status.  Logically translate an EOF card
;  959     1063  4         ! to an EOJ command.
;  960     1064  4         !
;  961     1065  4         $WAITFR(EFN=K_EFN_B);
;  962     1066  4         IF .CARD_IOSB_B[0] EQL SS$_ENDOFFILE
;  963     1067  4         THEN
;  964     1068  5             BEGIN
;  965     1069  5             INPUT_RAB[RAB$L_STS] = SS$_NORMAL;
;  966     1070  5             INPUT_RAB[RAB$W_RSZ] = %CHARCOUNT('$ EOJ');
;  967     1071  5             INPUT_RAB[RAB$L_RBF] = UPLIT BYTE('$ EOJ');
;  968     1072  5             END
;  969     1073  4         ELSE IF NOT .CARD_IOSB_B[0]
;  970     1074  4         THEN
;  971     1075  4             FILE_ERROR(
;  972     1076  4                 INPSMB$_FACILITY^16 + SHR$_READERR + STS$K_SEVERE,
;  973     1077  4                 INPUT_FAB,
;  974     1078  4                 .CARD_IOSB_B[0])
```

```
;  975          1079  4          ELSE
;  976          1080  5              BEGIN
;  977          1081  5              INPUT_RAB[RAB$L_STS] = .CARD_IOSB_B[0];
;  978          1082  5              INPUT_RAB[RAB$W_RSZ] = .CARD_IOSB_B[1];
;  979          1083  5              INPUT_RAB[RAB$L_RBF] = INPUT_UBF + 80;
;  980          1084  4              END;
;  981          1085  4
;  982          1086  4
;  983          1087  4          ! Start a read in the first buffer.
;  984          1088  4          !
;  985        P 1089  4          STATUS = $QIO(
;  986        P 1090  4              EFN=K_EFN_A,
;  987        P 1091  4              FUNC=IO$_READLBLK,
;  988        P 1092  4              CHAN=.CARD_CHANNEL,
;  989        P 1093  4              IOSB=CARD_IOSB_A,
;  990        P 1094  4              P1=INPUT_UBF,
;  991        P 1095  4              P2=80);
;  992          1096  4          FLAGS[V_SECOND_BUFFER] = FALSE;
;  993          1097  4          END
;  994          1098  3      ELSE
;  995          1099  4          BEGIN
;  996          1100  4
;  997          1101  4          ! The first buffer had the pending read.  Wait for it to complete
;  998          1102  4          ! and examine the IOSB for status.  Logically translate an EOF card
;  999          1103  4          ! to an EOJ command.
; 1000          1104  4          !
; 1001          1105  4          $WAITFR(EFN=K_EFN_A);
; 1002          1106  4          IF .CARD_IOSB_A[0] EQL SS$_ENDOFFILE
; 1003          1107  4          THEN
; 1004          1108  5              BEGIN
; 1005          1109  5              INPUT_RAB[RAB$L_STS] = SS$_NORMAL;
; 1006          1110  5              INPUT_RAB[RAB$W_RSZ] = %CHARCOUNT('$ EOJ');
; 1007          1111  5              INPUT_RAB[RAB$L_RBF] = UPLIT BYTE('$ EOJ');
; 1008          1112  5              END
; 1009          1113  4          ELSE IF NOT .CARD_IOSB_A[0]
; 1010          1114  4          THEN
; 1011          1115  4              FILE_ERROR(
; 1012          1116  4                  INPSMB$_FACILITY^16 + SHR$_READERR + STS$K_SEVERE,
; 1013          1117  4                  INPUT_FAB,
; 1014          1118  4                  .CARD_IOSB_A[0])
; 1015          1119  4          ELSE
; 1016          1120  5              BEGIN
; 1017          1121  5              INPUT_RAB[RAB$L_STS] = .CARD_IOSB_A[0];
; 1018          1122  5              INPUT_RAB[RAB$W_RSZ] = .CARD_IOSB_A[1];
; 1019          1123  5              INPUT_RAB[RAB$L_RBF] = INPUT_UBF;
; 1020          1124  4              END;
; 1021          1125  4
; 1022          1126  4
; 1023          1127  4          ! Start a read in the second buffer.
; 1024          1128  4          !
; 1025        P 1129  4          STATUS = $QIO(
; 1026        P 1130  4              EFN=K_EFN_B,
; 1027        P 1131  4              FUNC=IO$_READLBLK,
; 1028        P 1132  4              CHAN=.CARD_CHANNEL,
; 1029        P 1133  4              IOSB=CARD_IOSB_B,
; 1030        P 1134  4              P1=INPUT_UBF + 80,
; 1031          1135  4              P2=80);
```

```
;  1032          1136  4          FLAGS[V_SECOND_BUFFER] = TRUE;
;  1033          1137  3          END;
;  1034          1138  3
;  1035          1139  3
;  1036          1140  3      ! Check status of the $QIO.
;  1037          1141  3      !
;  1038          1142  3      IF NOT .STATUS
;  1039          1143  3      THEN
;  1040          1144  3          FILE_ERROR(
;  1041          1145  3              INPSMB$_FACILITY^16 + SHR$_READERR + STS$K_SEVERE,
;  1042          1146  3              INPUT_FAB,
;  1043          1147  3              .STATUS);
;  1044          1148  3
;  1045          1149  3
;  1046          1150  3      ! Note that an input operation has completed, for the periodic timer.
;  1047          1151  3      !
;  1048          1152  3      INPUT_COMPLETIONS = .INPUT_COMPLETIONS + 1;
;  1049          1153  3      END
;  1050          1154  2  ELSE
;  1051          1155  3      BEGIN
;  1052          1156  4      IF NOT $GET(RAB=INPUT_RAB)
;  1053          1157  3      THEN
;  1054          1158  3          IF .INPUT_RAB[RAB$L_STS] NEQ RMS$_EOF
;  1055          1159  3          THEN
;  1056          1160  3              FILE_ERROR(
;  1057          1161  3                  INPSMB$_FACILITY^16 + SHR$_READERR + STS$K_SEVERE,
;  1058          1162  3                  INPUT_FAB,
;  1059          1163  3                  .INPUT_RAB[RAB$L_STS], .INPUT_RAB[RAB$L_STV]);
;  1060          1164  2      END;
;  1061          1165  2
;  1062          1166  2
;  1063          1167  2  .INPUT_RAB[RAB$L_STS]
;  1064          1168  1  END;


                                4A 4F 45 20 24  008CF P.ABY:  .ASCII  \$ EOJ\                                  ;
                                4A 4F 45 20 24  008D4 P.ABZ:  .ASCII  \$ EOJ\                                  ;

                                                              .EXTRN  SYS$WAITFR, SYS$GET

                                              003C 00000 GET_RECORD:
                                                              .WORD   Save R2,R3,R4,R5                       ; 1022
                                55 00000000G   00  9E 00002         MOVAB   SYS$QIO, R5
                                54     0000V   CF  9E 00009         MOVAB   FILE_ERROR, R4
                                53 00000000G   00  9E 0000E         MOVAB   SYS$WAITFR, R3
                                52     0000'   CF  9E 00015         MOVAB   INPUT_RAB+8, R2
                                         A4    A2  B5 0001A         TSTW    CARD_CHANNEL                     ; 1050
                                               03  12 0001D         BNEQ    1$
                                           00E6  31 0001F         BRW     11$
                          65     04E0   C2        01  E1 00022 1$:  BBC     #1, FLAGS, 5$                    ; 1057
                                               02  DD 00028         PUSHL   #2                               ; 1065
                                63             01  FB 0002A         CALLS   #1, SYS$WAITFR
                                50     04F0    C2  3C 0002D         MOVZWL  CARD_IOSB_B, R0                  ; 1066
                                0870   8F              50  B1 00032         CMPW    R0, #2160
                                               0E  12 00037         BNEQ    2$
                                62             01  D0 00039         MOVL    #1, INPUT_RAB+8                  ; 1069
```

```
                1A  A2              05  B0  0003C          MOVW    #5, INPUT_RAB+34                        ;1070
                20  A2        B3    AF  9E  00040          MOVAB   P.ABY, INPUT_RAB+40                     ;1071
                                   22  11  00045          BRB     4$                                      ;1066
                    10             50  E8  00047  2$:      BLBS    R0, 3$                                  ;1073
                                   50  DD  0004A          PUSHL   R0                                      ;1078
                                A8 A2  9F  0004C          PUSHAB  INPUT_FAB                               ;1075
                        00000000* 8F   DD  0004F          PUSHL   #<<<INPSMB$_FACILITY@16>+4272>+4>       ;1076
                            64         03  FB  00055          CALLS   #3, FILE_ERROR
                                       0F  11  00058          BRB     4$                                      ;1075
                            62         50  D0  0005A  3$:      MOVL    R0, INPUT_RAB+8                         ;1081
                1A  A2          04F2   C2  B0  0005D          MOVW    CARD_IOSB_B+2, INPUT_RAB+34             ;1082
                20  A2          01EC   C2  9E  00063          MOVAB   INPUT_UBF+80, INPUT_RAB+40              ;1083
                                       7E  7C  00069  4$:      CLRQ    -(SP)                                   ;1095
                                       7E  7C  0006B          CLRQ    -(SP)
                            7E      50 8F  9A  0006D          MOVZBL  #80, -(SP)
                                    019C   C2  9F  00071          PUSHAB  INPUT_UBF
                                       7E  7C  00075          CLRQ    -(SP)
                                    04E8   C2  9F  00077          PUSHAB  CARD_IOSB_A
                                       21  DD  0007B          PUSHL   #33
                            7E      A4 A2  3C  0007D          MOVZWL  CARD_CHANNEL, -(SP)
                                       01  DD  00081          PUSHL   #1
                            65         0C  FB  00083          CALLS   #12, SYS$QIO
                    04E0    C2         02  8A  00086          BICB2   #2, FLAGS                               ;1096
                                       64  11  0008B          BRB     9$                                      ;1057
                                       01  DD  0008D  5$:      PUSHL   #1                                      ;1105
                            63         01  FB  0008F          CALLS   #1, SYS$WAITFR
                            50      04E8   C2  3C  00092          MOVZWL  CARD_IOSB_A, R0                         ;1106
                    0870    8F         50  B1  00097          CMPW    R0, #2160
                                       0F  12  0009C          BNEQ    6$
                            62         01  D0  0009E          MOVL    #1, INPUT_RAB+8                         ;1109
                1A  A2                 05  B0  000A1          MOVW    #5, INPUT_RAB+34                        ;1110
                20  A2          FF52   CF  9E  000A5          MOVAB   P.ABZ, INPUT_RAB+40                     ;1111
                                       22  11  000AB          BRB     8$                                      ;1106
                    10                 50  E8  000AD  6$:      BLBS    R0, 7$                                  ;1113
                                       50  DD  000B0          PUSHL   R0                                      ;1118
                                    A8 A2  9F  000B2          PUSHAB  INPUT_FAB                               ;1115
                            00000000* 8F   DD  000B5          PUSHL   #<<<INPSMB$_FACILITY@16>+4272>+4>       ;1116
                                64         03  FB  000BB          CALLS   #3, FILE_ERROR
                                       0F  11  000BE          BRB     8$                                      ;1115
                            62         50  D0  000C0  7$:      MOVL    R0, INPUT_RAB+8                         ;1121
                1A  A2          04EA   C2  B0  000C3          MOVW    CARD_IOSB_A+2, INPUT_RAB+34             ;1122
                20  A2          019C   C2  9E  000C9          MOVAB   INPUT_UBF, INPUT_RAB+40                 ;1123
                                       7E  7C  000CF  8$:      CLRQ    -(SP)                                   ;1135
                                       7E  7C  000D1          CLRQ    -(SP)
                            7E      50 8F  9A  000D3          MOVZBL  #80, -(SP)
                                    01EC   C2  9F  000D7          PUSHAB  INPUT_UBF+80
                                       7E  7C  000DB          CLRQ    -(SP)
                                    04F0   C2  9F  000DD          PUSHAB  CARD_IOSB_B
                                       21  DD  000E1          PUSHL   #33
                            7E      A4 A2  3C  000E3          MOVZWL  CARD_CHANNEL, -(SP)
                                       02  DD  000E7          PUSHL   #2
                            65         0C  FB  000E9          CALLS   #12, SYS$QIO
                    04E0    C2         02  88  000EC          BISB2   #2, FLAGS                               ;1136
                            0E         50  E8  000F1  9$:      BLBS    STATUS, 10$                             ;1142
                                       50  DD  000F4          PUSHL   STATUS                                  ;1147
                                    A8 A2  9F  000F6          PUSHAB  INPUT_FAB                               ;1144
                            00000000* 8F   DD  000F9          PUSHL   #<<<INPSMB$_FACILITY@16>+4272>+4>       ;1145
```

```
                          64              03  FB  000FF             CALLS   #3, FILE_ERROR                                   ;
                                    04E4  C2  D6  00102  10$:       INCL    INPUT_COMPLETIONS                               ; 1152
                                          25  11  00106             BRB     12$                                             ; 1050
                                      F8  A2  9F  00108  11$:       PUSHAB  INPUT_RAB                                       ; 1156
                00000000G             00  01  FB  0010B             CALLS   #1, SYS$GET                                     ;
                                      18  50  E8  00112             BLBS    R0, 12$                                         ;
                0001827A              8F  62  D1  00115             CMPL    INPUT_RAB+8, #98938                             ; 1158
                                          0F  13  0011C             BEQL    12$                                             ;
                                      7E  62  7D  0011E             MOVQ    INPUT_RAB+8, -(SP)                              ; 1163
                                        A8  A2  9F  00121           PUSHAB  INPUT_FAB                                       ; 1160
                                00000000*   8F  DD  00124           PUSHL   #<<<INPSMB$_FACILITY@16>+4272>+4>               ; 1161
                                      64        04  FB  0012A       CALLS   #4, FILE_ERROR                                  ;
                                      50        62  D0  0012D  12$: MOVL    INPUT_RAB+8, R0                                 ; 1168
                                                04  00130         RET                                                     ;

; Routine Size:  305 bytes,    Routine Base:  CODE + 08D9
```

```
;  1066     1169  1 ROUTINE IDENTIFY_COMMAND_VERB(PASSWORD,LINE_DESC)=
;  1067     1170  1
;  1068     1171  1 !++
;  1069     1172  1 !
;  1070     1173  1 ! FUNCTIONAL DESCRIPTION:
;  1071     1174  1 !       This routine identifies a record that contains a valid JOB, EOJ, or
;  1072     1175  1 !       PASSWORD command verb.
;  1073     1176  1 !
;  1074     1177  1 ! INPUT PARAMETERS:
;  1075     1178  1 !       PASSWORD        - True if a PASSWORD command is valid.
;  1076     1179  1 !       LINE_DESC       - Address of a quadword that receives a descriptor for
;  1077     1180  1 !                         the portion of the record following the dollar sign,
;  1078     1181  1 !                         if the routine value is true.
;  1079     1182  1 !
;  1080     1183  1 ! IMPLICIT INPUTS:
;  1081     1184  1 !       INPUT_RAB       - Describes the current record.
;  1082     1185  1 !
;  1083     1186  1 ! OUTPUT PARAMETERS:
;  1084     1187  1 !       NONE
;  1085     1188  1 !
;  1086     1189  1 ! IMPLICIT OUTPUTS:
;  1087     1190  1 !       NONE
;  1088     1191  1 !
;  1089     1192  1 ! ROUTINE VALUE:
;  1090     1193  1 !       K_NONE if no significant verb (false value).
;  1091     1194  1 !       K_JOB, K_EOJ, K_PASSWORD if recognized (true value).
;  1092     1195  1 !
;  1093     1196  1 ! SIDE EFFECTS:
;  1094     1197  1 !       NONE
;  1095     1198  1 !
;  1096     1199  1 !--
;  1097     1200  1
;  1098     1201  2 BEGIN
;  1099     1202  2 MAP
;  1100     1203  2         LINE_DESC:      REF BBLOCK;             ! Pointer to line descriptor
;  1101     1204  2 LOCAL
;  1102     1205  2         TPA_PARAM:      BBLOCK[TPA$K_LENGTH0],  ! TPARSE parameter block
;  1103     1206  2         UPCASE_BUFFER:  BBLOCK[%ALLOCATION(INPUT_UBF)];
;  1104     1207  2
;  1105     1208  2
;  1106     1209  2 ! Initialize TPARSE parameter block.
;  1107     1210  2 !
;  1108     1211  2 CH$FILL(0, %ALLOCATION(TPA_PARAM), TPA_PARAM);
;  1109     1212  2 TPA_PARAM[TPA$L_COUNT] = TPA$K_COUNT0;
;  1110     1213  2 TPA_PARAM[TPA$L_STRINGCNT] = .INPUT_RAB[RAB$W_RSZ];
;  1111     1214  2 TPA_PARAM[TPA$L_STRINGPTR] = .INPUT_RAB[RAB$L_RBF];
;  1112     1215  2
;  1113     1216  2
;  1114     1217  2 ! Scan the line for a leading dollar sign.
;  1115     1218  2 !
;  1116     1219  2 IF LIB$TPARSE(TPA_PARAM, DOLLAR_STATES, DOLLAR_KEYS)
;  1117     1220  2 THEN
;  1118     1221  3     BEGIN
;  1119     1222  3
;  1120     1223  3     ! Initialize the line descriptor to describe the portion of the line
;  1121     1224  3     ! following the leading dollar sign.
;  1122     1225  3     !
```

```
;  1123          1226  3     LINE_DESC[DSC$W_LENGTH] = .TPA_PARAM[TPA$L_STRINGCNT];
;  1124          1227  3     LINE_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  1125          1228  3     LINE_DESC[DSC$B_CLASS] = DSC$K_CLASS_S;
;  1126          1229  3     LINE_DESC[DSC$A_POINTER] = .TPA_PARAM[TPA$L_STRINGPTR];
;  1127          1230  3
;  1128          1231  3
;  1129          1232  3     ! Upcase the remaining portion of the line into the temporary buffer.
;  1130          1233  3     !
;  1131          1234  3     MOVTC(
;  1132          1235  3         TPA_PARAM[TPA$L_STRINGCNT], .TPA_PARAM[TPA$L_STRINGPTR],
;  1133          1236  3         %REF(0),
;  1134          1237  3         LIB$AB_UPCASE,
;  1135          1238  3         TPA_PARAM[TPA$L_STRINGCNT], UPCASE_BUFFER);
;  1136          1239  3     TPA_PARAM[TPA$L_STRINGPTR] = UPCASE_BUFFER;
;  1137          1240  3
;  1138          1241  3
;  1139          1242  3     ! Scan the line for an unabbreviated 'JOB'.
;  1140          1243  3     !
;  1141          1244  3     IF LIB$TPARSE(TPA_PARAM, JOB_STATES, JOB_KEYS)
;  1142          1245  3     THEN
;  1143          1246  3         RETURN K_JOB;
;  1144          1247  3
;  1145          1248  3
;  1146          1249  3     ! Scan the line for an unabbreviated 'EOJ'.
;  1147          1250  3     !
;  1148          1251  3     IF LIB$TPARSE(TPA_PARAM, EOJ_STATES, EOJ_KEYS)
;  1149          1252  3     THEN
;  1150          1253  3         RETURN K_EOJ;
;  1151          1254  3
;  1152          1255  3
;  1153          1256  3     ! If a PASSWORD command is valid, scan the line for a possibly abbreviated
;  1154          1257  3     ! 'PASSWORD'.
;  1155          1258  3     !
;  1156          1259  3     IF .PASSWORD
;  1157          1260  3     THEN
;  1158          1261  4         BEGIN
;  1159          1262  4         TPA_PARAM[TPA$V_ABBREV] = TRUE;
;  1160          1263  4         IF [IB$TPARSE(TPA_PARAM, PASSWORD_STATES, PASSWORD_KEYS)
;  1161          1264  4         THEN
;  1162          1265  4             RETURN K_PASSWORD;
;  1163          1266  3         END;
;  1164          1267  2     END;
;  1165          1268  2
;  1166          1269  2
;  1167          1270  2 K_NONE
;  1168          1271  1 END;


                                        007C 00000 IDENTIFY_COMMAND_VERB:
                                                   .WORD   Save R2,R3,R4,R5,R6                          ; 1169
                          56 00000000G   00  9E 00002      MOVAB   LIB$TPARSE, R6
                          5E     FF3C    CE  9E 00009      MOVAB   -196(SP), SP
          24              00      6E     00  2C 0000E      MOVC5   #0, (SP), #0, #36, TPA_PARAM                 ; 1211
                                  DC     AD     00013
```

```
                        DC   AD          08  D0  00015          MOVL    #8, TPA_PARAM                                   ; 1212
                        E4   AD    0000' CF  3C  00019          MOVZWL  INPUT_RAB+34, TPA_PARAM+8                       ; 1213
                        E8   AD    0000' CF  D0  0001F          MOVL    INPUT_RAB+40, TPA_PARAM+12                      ; 1214
                             0000V CF  9F  00025                PUSHAB  DOLLAR_KEYS                                     ; 1219
                             0000V CF  9F  00029                PUSHAB  DOLLAR_STATES
                               DC  AD  9F  0002D                PUSHAB  TPA_PARAM
                             66    03  FB  00030                CALLS   #3, LIB$TPARSE
                             6C    50  E9  00033                BLBC    R0, 3$
                             50    08  AC  D0  00036           MOVL    LINE_DESC, R0                                   ; 1226
                             60    E4  AD  B0  0003A           MOVW    TPA_PARAM+8, (R0)
                        02   A0  010E  8F  B0  0003E           MOVW    #270, 2(R0)                                     ; 1227
                        04   A0    E8  AD  D0  00044           MOVL    TPA_PARAM+12, 4(R0)                             ; 1229
00000000G  00     00    E8   BD    E4  AD  2E  00049           MOVTC   TPA_PARAM+8, @TPA_PARAM+12, #0, -               ; 1234
                             6E    E4  AD      00054                   LIB$AB_UPCASE, TPA_PARAM+8, UPCASE_BUFFER
                        E8   AD          6E  9E  00057           MOVAB   UPCASE_BUFFER, TPA_PARAM+12                     ; 1239
                             0000V CF  9F  0005B                PUSHAB  JOB_KEYS                                        ; 1244
                             0000V CF  9F  0005F                PUSHAB  JOB_STATES
                               DC  AD  9F  00063                PUSHAB  TPA_PARAM
                             66    03  FB  00066                CALLS   #3, LIB$TPARSE
                             04    50  E9  00069                BLBC    R0, 1$
                             50          01  D0  0006C           MOVL    #1, R0                                          ; 1246
                                         04  0006F           RET
                             0000V CF  9F  00070  1$:           PUSHAB  EOJ_KEYS                                        ; 1251
                             0000V CF  9F  00074                PUSHAB  EOJ_STATES
                               DC  AD  9F  00078                PUSHAB  TPA_PARAM
                             66    03  FB  0007B                CALLS   #3, LIB$TPARSE
                             04    50  E9  0007E                BLBC    R0, 2$
                             50          03  D0  00081           MOVL    #3, R0                                          ; 1253
                                         04  00084           RET
                             19    04    AC  E9  00085  2$:     BLBC    PASSWORD, 3$                                    ; 1259
                        E0   AD          02  88  00089          BISB2   #2, TPA_PARAM+4                                 ; 1262
                             0000V CF  9F  0008D                PUSHAB  PASSWORD_KEYS                                   ; 1263
                             0000V CF  9F  00091                PUSHAB  PASSWORD_STATES
                               DC  AD  9F  00095                PUSHAB  TPA_PARAM
                             66    03  FB  00098                CALLS   #3, LIB$TPARSE
                             04    50  E9  0009B                BLBC    R0, 3$
                             50          05  D0  0009E           MOVL    #5, R0                                          ; 1265
                                         04  000A1           RET
                                         50  D4  000A2  3$:     CLRL    R0                                              ; 1271
                                         04  000A4           RET

; Routine Size:  165 bytes,    Routine Base:  CODE + 0A0A
```

```
;  1170          1272  1 ROUTINE GET_LINE_CONTINUATION(GET_STR,PROMPT_STR,OUT_LEN)=
;  1171          1273  1
;  1172          1274  1 !++
;  1173          1275  1 !
;  1174          1276  1 ! FUNCTIONAL DESCRIPTION:
;  1175          1277  1 !       This routine is the continuation routine for the CLI$DCL_PARSE calls.
;  1176          1278  1 !
;  1177          1279  1 ! INPUT PARAMETERS:
;  1178          1280  1 !       As for LIB$GET_INPUT.
;  1179          1281  1 !
;  1180          1282  1 ! IMPLICIT INPUTS:
;  1181          1283  1 !       NONE
;  1182          1284  1 !
;  1183          1285  1 ! OUTPUT PARAMETERS:
;  1184          1286  1 !       NONE
;  1185          1287  1 !
;  1186          1288  1 ! IMPLICIT OUTPUTS:
;  1187          1289  1 !       NONE
;  1188          1290  1 !
;  1189          1291  1 ! ROUTINE VALUE:
;  1190          1292  1 !       As for LIB$GET_INPUT.
;  1191          1293  1 !
;  1192          1294  1 ! SIDE EFFECTS:
;  1193          1295  1 !       NONE
;  1194          1296  1 !
;  1195          1297  1 !--
;  1196          1298  1
;  1197          1299  2 BEGIN                \
;  1198          1300  2 MAP
;  1199          1301  2         GET_STR:        REF BBLOCK;             ! Pointer to descriptor
;  1200          1302  2 LOCAL
;  1201          1303  2         LINE_DESC:      BBLOCK[DSC$C_S_BLN],    ! Scratch descriptor for line
;  1202          1304  2         STATUS;                                 ! Status return
;  1203          1305  2
;  1204          1306  2
;  1205          1307  2 ! Get the next input line, propagating errors to CLI$DCL_PARSE.
;  1206          1308  2 !
;  1207          1309  2 STATUS = GET_RECORD();
;  1208          1310  2 IF NOT .STATUS THEN RETURN .STATUS;
;  1209          1311  2
;  1210          1312  2
;  1211          1313  2 ! Ensure that the continuation line is not a JOB command, so that an error in
;  1212          1314  2 ! a previous line cannot result in skipping a job.
;  1213          1315  2 !
;  1214          1316  2 CURRENT_COMMAND = IDENTIFY_COMMAND_VERB(FALSE, LINE_DESC);
;  1215          1317  2 IF .CURRENT_COMMAND EQL K_JOB THEN RETURN INPSMB$_INVCONT;
;  1216          1318  2
;  1217          1319  2
;  1218          1320  2 ! Copy the record back to DCL and set the return length.  This routine makes
;  1219          1321  2 ! the simplifying assumptions that DCL passes a static string and always
;  1220          1322  2 ! passes three parameters.
;  1221          1323  2 !
;  1222          1324  2 CH$COPY(
;  1223          1325  2     .INPUT_RAB[RAB$W_RSZ], .INPUT_RAB[RAB$L_RBF],
;  1224          1326  2     %C' ',
;  1225          1327  2     .GET_STR[DSC$W_LENGTH], .GET_STR[DSC$A_POINTER]);
;  1226          1328  2 (.OUT_LEN)<0,16> = .INPUT_RAB[RAB$W_RSZ];
```

```
;  1227          1329  2
;  1228          1330  2
;  1229          1331  2 ! Return success.
;  1230          1332  2 !
;  1231          1333  2 SS$_NORMAL
;  1232          1334  1 END;


                                                003C 00000 GET_LINE_CONTINUATION:
                                                           .WORD   Save R2,R3,R4,R5                                        ;  1272
                                 5E                 08  C2 00002                SUBL2   #8, SP                                  ;
                           FE20  CF                 00  FB 00005                CALLS   #0, GET_RECORD                          ;  1309
                                 35                 50  E9 0000A                BLBC    STATUS, 2$                              ;  1310
                                                    5E  DD 0000D                PUSHL   SP                                      ;  1316
                                                    7E  D4 0000F                CLRL    -(SP)                                   ;
                           FF45  CF                 02  FB 00011                CALLS   #2, IDENTIFY_COMMAND_VERB               ;
                           0000' CF                 50  D0 00016                MOVL    R0, CURRENT_COMMAND                     ;
                                 01          0000'  CF  D1 0001B                CMPL    CURRENT_COMMAND, #1                     ;  1317
                                                    08  12 00020                BNEQ    1$                                      ;
                                 50 00000000G       8F  D0 00022                MOVL    #INPSMB$_INVCONT, R0                    ;
                                                    04  00029                   RET                                             ;
                                 50            04   AC  D0 0002A 1$:           MOVL    GET_STR, R0                             ;  1327
          60            20     0000' DF      0000'  CF  2C 0002E                MOVC5   INPUT_RAB+34, @INPUT_RAB+40, #32, (R0), -  ;
                                                04   B0  00037                   @4(R0)                                          ;
                             0C  BC      0000'  CF  B0 00039                MOVW    INPUT_RAB+34, @OUT_LEN                  ;  1328
                                 50                 01  D0 0003F                MOVL    #1, R0                                  ;  1334
                                                    04  00042 2$:              RET                                             ;

; Routine Size:  67 bytes,    Routine Base:  CODE + 0AAF
```

```
;  1234     1335  1 ROUTINE TIMER_AST: NOVALUE=
;  1235     1336  1
;  1236     1337  1 !++
;  1237     1338  1 !
;  1238     1339  1 ! FUNCTIONAL DESCRIPTION:
;  1239     1340  1 !       This routine is entered on the expiration of the periodic timer to
;  1240     1341  1 !       determine if any input operations have completed in that interval,
;  1241     1342  1 !       and to exit the symbiont if appropriate.
;  1242     1343  1 !
;  1243     1344  1 ! INPUT PARAMETERS:
;  1244     1345  1 !       Standard AST routine parameters (not used).
;  1245     1346  1 !
;  1246     1347  1 ! IMPLICIT INPUTS:
;  1247     1348  1 !       NONE
;  1248     1349  1 !
;  1249     1350  1 ! OUTPUT PARAMETERS:
;  1250     1351  1 !       NONE
;  1251     1352  1 !
;  1252     1353  1 ! IMPLICIT OUTPUTS:
;  1253     1354  1 !       NONE
;  1254     1355  1 !
;  1255     1356  1 ! ROUTINE VALUE:
;  1256     1357  1 !       NONE
;  1257     1358  1 !
;  1258     1359  1 ! SIDE EFFECTS:
;  1259     1360  1 !       NONE
;  1260     1361  1 !
;  1261     1362  1 !--
;  1262     1363  1
;  1263     1364  2 BEGIN
;  1264     1365  2
;  1265     1366  2 ! If there have been no input completions since the last expiration of the timer
;  1266     1367  2 ! and we are not processing a job, exit the symbiont.
;  1267     1368  2 !
;  1268     1369  2 IF .INPUT_COMPLETIONS EQL 0 AND .OUTPUT_FAB[FAB$W_IFI] EQL 0
;  1269     1370  2 THEN
;  1270     1371  2     $EXIT(CODE=SS$_NORMAL);
;  1271     1372  2
;  1272     1373  2
;  1273     1374  2 ! Set up the next interval.
;  1274     1375  2 !
;  1275     1376  2 INPUT_COMPLETIONS = 0;
;  1276     1377  2 $SETIMR(DAYTIM=PERIODIC_INTERVAL, ASTADR=TIMER_AST);
;  1277     1378  1 END;


                                                .EXTRN  SYS$EXIT

                                0000 00000 TIMER_AST:
                                                .WORD   Save nothing                    ;  1335
                          0000'  CF  D5 00002           TSTL    INPUT_COMPLETIONS       ;  1369
                                 0F  12 00006           BNEQ    1$                      ;
                          0000'  CF  B5 00008           TSTW    OUTPUT_FAB+2            ;
                                 09  12 0000C           BNEQ    1$                      ;
                                 01  DD 0000E           PUSHL   #1                      ;  1371
                00000000G  00    01  FB 00010           CALLS   #1, SYS$EXIT            ;
```

```
                0000'  CF  D4 00017 1$:     CLRL    INPUT_COMPLETIONS                ; 1376
                       7E  D4 0001B         CLRL    -(SP)                            ; 1377
                    E0 AF  9F 0001D         PUSHAB  TIMER_AST                        ;
                F4EA   CF  9F 00020         PUSHAB  PERIODIC_INTERVAL                ;
                       7E  D4 00024         CLRL    -(SP)                            ;
      00000000G  00        04  FB 00026     CALLS   #4, SYS$SETIMR                   ;
                           04  0002D         RET                                      ; 1378

; Routine Size:  46 bytes,    Routine Base:  CODE + 0AF2
```

```
;  1279     1379  1 ROUTINE FILE_ERROR(MESSAGE,FAB,EXTRA1,EXTRA2): NOVALUE=
;  1280     1380  1
;  1281     1381  1 !++
;  1282     1382  1 !
;  1283     1383  1 ! FUNCTIONAL DESCRIPTION:
;  1284     1384  1 !       This routine signals a file-related message.
;  1285     1385  1 !
;  1286     1386  1 ! INPUT PARAMETERS:
;  1287     1387  1 !       MESSAGE         - Message code for first message
;  1288     1388  1 !       FAB             - Pointer to FAB, from which file name
;  1289     1389  1 !                         will be obtained
;  1290     1390  1 !       Up to two additional input parameters are additional messages.
;  1291     1391  1 !
;  1292     1392  1 ! IMPLICIT INPUTS:
;  1293     1393  1 !       NONE
;  1294     1394  1 !
;  1295     1395  1 ! OUTPUT PARAMETERS:
;  1296     1396  1 !       NONE
;  1297     1397  1 !
;  1298     1398  1 ! IMPLICIT OUTPUTS:
;  1299     1399  1 !       NONE
;  1300     1400  1 !
;  1301     1401  1 ! ROUTINE VALUE:
;  1302     1402  1 !       NONE
;  1303     1403  1 !
;  1304     1404  1 ! SIDE EFFECTS:
;  1305     1405  1 !       The messages are signalled.
;  1306     1406  1 !
;  1307     1407  1 !--
;  1308     1408  1
;  1309     1409  2 BEGIN
;  1310     1410  2 MAP
;  1311     1411  2         FAB:            REF BBLOCK;     ! Pointer to FAB
;  1312     1412  2 LOCAL
;  1313     1413  2         NAM:            REF BBLOCK,     ! Pointer to NAM block
;  1314     1414  2         DESC:           VECTOR[2],      ! Descriptor for file name
;  1315     1415  2         PARAM:          VECTOR[6];      ! Signal parameter list
;  1316     1416  2 BUILTIN
;  1317     1417  2         ACTUALCOUNT,
;  1318     1418  2         CALLG;
;  1319     1419  2
;  1320     1420  2
;  1321     1421  2 ! Establish the file name to be printed.  The resultant string, expanded
;  1322     1422  2 ! string, and filename string are examined in that order, and the first
;  1323     1423  2 ! one that is not null is used.
;  1324     1424  2 !
;  1325     1425  2 NAM = .FAB[FAB$L_NAM];
;  1326     1426  2 IF .NAM[NAM$B_RS[] NEQ 0
;  1327     1427  2 THEN
;  1328     1428  3     BEGIN
;  1329     1429  3     DESC[0] = .NAM[NAM$B_RSL];
;  1330     1430  3     DESC[1] = .NAM[NAM$L_RSA];
;  1331     1431  3     END
;  1332     1432  2 ELSE IF .NAM[NAM$B_ESL] NEQ 0
;  1333     1433  2 THEN
;  1334     1434  3     BEGIN
;  1335     1435  3     DESC[0] = .NAM[NAM$B_ESL];
```

```
;  1336          1436  3      DESC[1] = .NAM[NAM$L_ESA];
;  1337          1437  3      END
;  1338          1438  2  ELSE
;  1339          1439  3      BEGIN
;  1340          1440  3      DESC[0] = .FAB[FAB$B_FNS];
;  1341          1441  3      DESC[1] = .FAB[FAB$L_FNA];
;  1342          1442  2      END;
;  1343          1443  2
;  1344          1444  2
;  1345          1445  2  ! Initialize the signal parameter list.
;  1346          1446  2  !
;  1347          1447  2  PARAM[0] = 3;                          ! Parameter count
;  1348          1448  2  PARAM[1] = .MESSAGE;                   ! First message code
;  1349          1449  2  PARAM[2] = 1;                          ! FAO argument count
;  1350          1450  2  PARAM[3] = DESC;                       ! Filename descriptor
;  1351          1451  2  IF ACTUALCOUNT() GEQ 3
;  1352          1452  2  THEN
;  1353          1453  3      BEGIN
;  1354          1454  3      PARAM[0] = .PARAM[0] + 1;          ! Increment parameter count
;  1355          1455  3      PARAM[4] = .EXTRA1;                ! Next message code
;  1356          1456  2      END;
;  1357          1457  2  IF ACTUALCOUNT() GEQ 4
;  1358          1458  2  THEN
;  1359          1459  3      BEGIN
;  1360          1460  3      PARAM[0] = .PARAM[0] + 1;          ! Increment parameter count
;  1361          1461  3      PARAM[5] = .EXTRA2;                ! Next message code
;  1362          1462  2      END;
;  1363          1463  2
;  1364          1464  2
;  1365          1465  2  ! Finally, signal the messages.
;  1366          1466  2  !
;  1367          1467  2  CALLG(PARAM, LIB$SIGNAL);
;  1368          1468  1  END;


                                0000 00000 FILE_ERROR:
                                           .WORD   Save nothing                    ;  1379
                           5E        20  C2 00002                SUBL2   #32, SP
                           51        AC  D0 00005                MOVL    FAB, R1 ;  1425
                           50  28    A1  D0 00009                MOVL    40(R1), NAM
                               03    A0  95 0000D                TSTB    3(NAM)  ;  1426
                                     0C  13 00010                BEQL    1$
                     18    AE    03    A0  9A 00012                MOVZBL  3(NAM), DESC    ;  1429
                     1C    AE    04    A0  D0 00017                MOVL    4(NAM), DESC+4  ;  1430
                                     1B  11 0001C                BRB     3$              ;  1426
                               0B    A0  95 0001E 1$:            TSTB    11(NAM) ;  1432
                                     0C  13 00021                BEQL    2$
                     18    AE    0B    A0  9A 00023                MOVZBL  11(NAM), DESC   ;  1435
                     1C    AE    0C    A0  D0 00028                MOVL    12(NAM), DESC+4 ;  1436
                                     0A  11 0002D                BRB     3$              ;  1432
                     18    AE    34    A1  9A 0002F 2$:            MOVZBL  52(R1), DESC    ;  1440
                     1C    AE    2C    A1  D0 00034                MOVL    44(R1), DESC+4  ;  1441
                           6E          03  D0 00039 3$:            MOVL    #3, PARAM       ;  1447
                     04    AE    04    AC  D0 0003C                MOVL    MESSAGE, PARAM+4        ;  1448
```

```
                        08   AE          01   D0 00041          MOVL    #1, PARAM+8                     ; 1449
                        0C   AE     18   AE   9E 00045          MOVAB   DESC, PARAM+12                  ; 1450
                             03     6C   91 0004A              CMPB    (AP), #3                        ; 1451
                                    07   1F 0004D              BLSSU   4$                              ;
                                    6E   D6 0004F              INCL    PARAM                           ; 1454
                        10   AE     0C   AC   D0 00051         MOVL    EXTRA1, PARAM+16                ; 1455
                             04     6C   91 00056 4$:          CMPB    (AP), #4                        ; 1457
                                    07   1F 00059              BLSSU   5$                              ;
                                    6E   D6 0005B              INCL    PARAM                           ; 1460
                        14   AE     10   AC   D0 0005D         MOVL    EXTRA2, PARAM+20                ; 1461
                00000000G    00          6E   FA 00062 5$:     CALLG   PARAM, LIB$SIGNAL               ; 1467
                                         04 00069              RET                                     ; 1468

; Routine Size:  106 bytes,    Routine Base:  CODE + 0B20
```

```
;  1370          1469  1 ROUTINE MAIN_HANDLER_ACTION(MSG_DESC)=
;  1371          1470  1
;  1372          1471  1 !++
;  1373          1472  1 !
;  1374          1473  1 ! FUNCTIONAL DESCRIPTION:
;  1375          1474  1 !       This is an action routine for the $PUTMSG that issues a signalled
;  1376          1475  1 !       message to the system console.  It writes the record to the operator
;  1377          1476  1 !       via OPCOM or via broadcast.
;  1378          1477  1 !
;  1379          1478  1 ! INPUT PARAMETERS:
;  1380          1479  1 !       MSG_DESC        - Descriptor for message.
;  1381          1480  1 !
;  1382          1481  1 ! IMPLICIT INPUTS:
;  1383          1482  1 !       NONE
;  1384          1483  1 !
;  1385          1484  1 ! OUTPUT PARAMETERS:
;  1386          1485  1 !       NONE
;  1387          1486  1 !
;  1388          1487  1 ! IMPLICIT OUTPUTS:
;  1389          1488  1 !       NONE
;  1390          1489  1 !
;  1391          1490  1 ! ROUTINE VALUE:
;  1392          1491  1 !       FALSE, to signal $PUTMSG not to write the message.
;  1393          1492  1 !
;  1394          1493  1 ! SIDE EFFECTS:
;  1395          1494  1 !       NONE
;  1396          1495  1 !
;  1397          1496  1 !--
;  1398          1497  1
;  1399          1498  2 BEGIN
;  1400          1499  2 MAP
;  1401          1500  2         MSG_DESC:       REF BBLOCK;     ! Descriptor for message text
;  1402          1501  2 LOCAL
;  1403          1502  2         LENGTH:         WORD,           ! Length of message, minimized
;  1404          1503  2         OPC_BUFFER:     BBLOCK[$BYTEOFFSET(OPC$L_MS_TEXT) + 132],
;  1405          1504  2                                         ! Buffer for OPCOM message
;  1406          1505  2         OPC_DESC:       VECTOR[2],      ! Descriptor for message buffer
;  1407          1506  2         STATUS;                         ! Status return
;  1408          1507  2
;  1409          1508  2
;  1410          1509  2 ! Set up the OPCOM message buffer.
;  1411          1510  2 !
;  1412          1511  2 OPC_BUFFER[OPC$B_MS_TYPE] = OPC$_RQ_RQST;
;  1413          1512  2 OPC_BUFFER[OPC$B_MS_TARGET] = OPC$M_NM_CARDS;
;  1414          1513  2 OPC_BUFFER[OPC$W_MS_STATUS] = 0;
;  1415          1514  2 OPC_BUFFER[OPC$L_MS_RQSTID] = 0;
;  1416          1515  2 LENGTH = .MSG_DESC[DSC$W_LENGTH];
;  1417          1516  2 IF .LENGTH GTRU 132 THEN LENGTH = 132;
;  1418          1517  2 CH$MOVE(.LENGTH, .MSG_DESC[DSC$A_POINTER], OPC_BUFFER[OPC$L_MS_TEXT]);
;  1419          1518  2 OPC_DESC[0] = $BYTEOFFSET(OPC$L_MS_TEXT) + .LENGTH;
;  1420          1519  2 OPC_DESC[1] = OPC_BUFFER;
;  1421          1520  2
;  1422          1521  2
;  1423          1522  2 ! Try to send the message by OPCOM.  If this fails, send a broadcast to the
;  1424          1523  2 ! system console.
;  1425          1524  2 !
;  1426          1525  2 STATUS = $SNDOPR(MSGBUF=OPC_DESC);
```

```
;  1427          1526  2 IF NOT .STATUS OR .STATUS EQL OPC$_NOPERATOR
;  1428          1527  2 THEN
;  1429          1528  2     $BRDCST(MSGBUF=.MSG_DESC, DEVNAM=$DESCRIPTOR('_OPA0:'));
;  1430          1529  2
;  1431          1530  2
;  1432          1531  2 ! Return FALSE, to signal $PUTMSG not to write the message.
;  1433          1532  2 !
;  1434          1533  2 FALSE
;  1435          1534  1 END;


                                3A  30  41  50  4F  5F  00B8A P.ACB:   .ASCII  \_OPA0:\                                  ;
                                              00000006  00B90 P.ACA:   .LONG   6                                          ;
                                              00000000'  00B94         .ADDRESS P.ACB                                    ;

                                                                       .EXTRN  SYS$SNDOPR, SYS$BRDCST

                                                    00FC 00000 MAIN_HANDLER_ACTION:
                                                               .WORD   Save R2,R3,R4,R5,R6,R7                           ;  1469
                                    5E      FF6C    CE  9E 00002       MOVAB   -148(SP), SP                             ;
                                08  AE      2003    8F  3C 00007       MOVZWL  #8195, OPC_BUFFER                        ;  1511
                                    0C      AE      D4  0000D          CLRL    OPC_BUFFER+4                             ;  1514
                                    57      04      AC  D0 00010       MOVL    MSG_DESC, R7                             ;  1515
                                    56              67  B0 00014       MOVW    (R7), LENGTH                             ;
                              0084  8F              56  B1 00017       CMPW    LENGTH, #132                             ;  1516
                                                    04  1B 0001C       BLEQU   1$                                       ;
                                    56      84      8F  9B 0001E       MOVZBW  #132, LENGTH                             ;
                            10  AE      04  B7              56  28 00022 1$:   MOVC3   LENGTH, @4(R7), OPC_BUFFER+8    ;  1517
                                    6E              56  3C 00028       MOVZWL  LENGTH, OPC_DESC                         ;  1518
                                    6E              08  C0 0002B       ADDL2   #8, OPC_DESC                             ;
                                04  AE      08      AE  9E 0002E       MOVAB   OPC_BUFFER, OPC_DESC+4                   ;  1519
                                    7E      D4  00033                  CLRL    -(SP)                                    ;  1525
                                    04      AE  9F 00035               PUSHAB  OPC_DESC                                 ;
                        00000000G   00          02  FB 00038          CALLS   #2, SYS$SNDOPR                           ;
                                    09          50  E9 0003F          BLBC    STATUS, 2$                               ;  1526
                        00058061    8F          50  D1 00042          CMPL    STATUS, #360545                          ;
                                                10  12 00049          BNEQ    3$                                       ;
                                                20  DD 0004B 2$:      PUSHL   #32                                      ;  1528
                                                7E  D4 0004D           CLRL    -(SP)                                    ;
                                        A6      AF  9F 0004F           PUSHAB  P.ACA                                    ;
                                                57  DD 00052           PUSHL   R7                                       ;
                        00000000G   00          04  FB 00054          CALLS   #4, SYS$BRDCST                           ;
                                                50  D4 0005B 3$:      CLRL    R0                                       ;  1534
                                                04  0005D             RET                                              ;

; Routine Size:  94 bytes,    Routine Base:  CODE + 0B98
```

```
;  1437     1535  1 ROUTINE MAIN_HANDLER(SIG,MCH)=
;  1438     1536  1
;  1439     1537  1 !++
;  1440     1538  1 !
;  1441     1539  1 ! FUNCTIONAL DESCRIPTION:
;  1442     1540  1 !       This routine is the condition handler for the main routine.  It
;  1443     1541  1 !       intercepts signals and writes the message to the operator.
;  1444     1542  1 !
;  1445     1543  1 ! INPUT PARAMETERS:
;  1446     1544  1 !       Standard VMS condition handler parameters.
;  1447     1545  1 !
;  1448     1546  1 ! IMPLICIT INPUTS:
;  1449     1547  1 !       NONE
;  1450     1548  1 !
;  1451     1549  1 ! OUTPUT PARAMETERS:
;  1452     1550  1 !       NONE
;  1453     1551  1 !
;  1454     1552  1 ! IMPLICIT OUTPUTS:
;  1455     1553  1 !       NONE
;  1456     1554  1 !
;  1457     1555  1 ! ROUTINE VALUE:
;  1458     1556  1 !       SS$_CONTINUE
;  1459     1557  1 !
;  1460     1558  1 ! SIDE EFFECTS:
;  1461     1559  1 !       If the condition is fatal, the image exits.
;  1462     1560  1 !
;  1463     1561  1 !--
;  1464     1562  1
;  1465     1563  2 BEGIN
;  1466     1564  2 MAP
;  1467     1565  2         SIG:            REF BBLOCK,     ! Signal parameters
;  1468     1566  2         MCH:            REF BBLOCK;     ! Mechanism parameters
;  1469     1567  2 LOCAL
;  1470     1568  2         DESC:           VECTOR[2],      ! Descriptor for JOB command
;  1471     1569  2         MSGVEC:         VECTOR[4];      ! $PUTMSG parameter vector
;  1472     1570  2
;  1473     1571  2
;  1474     1572  2 ! Print the JOB command that incurred the error, if any.
;  1475     1573  2 !
;  1476     1574  2 IF .JOB_LENGTH NEQ 0
;  1477     1575  2 THEN
;  1478     1576  3     BEGIN
;  1479     1577  3     DESC[0] = .JOB_LENGTH;
;  1480     1578  3     DESC[1] = JOB_BUFFER;
;  1481     1579  3     WHILE .DESC[0] GTR 0 DO
;  1482     1580  4         BEGIN
;  1483     1581  4         IF CH$RCHAR(.DESC[1] + .DESC[0] - 1) NEQ %C' ' THEN EXITLOOP;
;  1484     1582  4         DESC[0] = .DESC[0] - 1;
;  1485     1583  3         END;
;  1486     1584  3     MSGVEC[0] = 3;
;  1487     1585  3     MSGVEC[1] = INPSMB$_JOBCARD;
;  1488     1586  3     MSGVEC[2] = 1;
;  1489     1587  3     MSGVEC[3] = DESC;
;  1490     1588  3     $PUTMSG(MSGVEC=MSGVEC, ACTRTN=.PUTMSG_ACTION_ROUTINE);
;  1491     1589  2     END;
;  1492     1590  2
;  1493     1591  2
```

```
; 1494      1592 2 ! Adjust the signal parameter count to remove the PC and PSL, and call $PUTMSG
; 1495      1593 2 ! to issue the message.
; 1496      1594 2 !
; 1497      1595 2 SIG[CHF$L_SIG_ARGS] = .SIG[CHF$L_SIG_ARGS] - 2;
; 1498      1596 2 $PUTMSG(MSGVEC=.SIG, ACTRTN=.PUTMSG_ACTION_ROUTINE);
; 1499      1597 2
; 1500      1598 2
; 1501      1599 2 ! If the exception was fatal, exit the image.  Otherwise, continue.
; 1502      1600 2 !
; 1503      1601 2 IF .BBLOCK[SIG[CHF$L_SIG_NAME], STS$V_SEVERITY] EQL STS$K_SEVERE
; 1504      1602 2 THEN
; 1505      1603 2     $EXIT(CODE=.SIG[CHF$L_SIG_NAME] OR STS$M_INHIB_MSG);
; 1506      1604 2
; 1507      1605 2
; 1508      1606 2 SS$_CONTINUE
; 1509      1607 1 END;
```

```
                                                          .EXTRN  SYS$PUTMSG

                                         001C 00000 MAIN_HANDLER:
                                                          .WORD   Save R2,R3,R4                            ; 1535
                   54      0000'    CF   9E 00002         MOVAB   PUTMSG_ACTION_ROUTINE, R4
                   53 00000000G     00   9E 00007         MOVAB   SYS$PUTMSG, R3
                   5E               18   C2 0000E         SUBL2   #24, SP
                   50          AC   A4   D0 00011         MOVL    JOB_LENGTH, R0                           ; 1574
                                    3D   13 00015         BEQL    3$
              10   AE               50   D0 00017         MOVL    R0, DESC                                 ; 1577
              14   AE          B0   A4   9E 0001B         MOVAB   JOB_BUFFER, DESC+4                       ; 1578
                               10   AE   D5 00020 1$:     TSTL    DESC                                     ; 1579
                                    11   15 00023         BLEQ    2$
     50       14   AE          10   AE   C1 00025         ADDL3   DESC, DESC+4, R0                         ; 1581
                   20          FF   A0   91 0002B         CMPB    -1(R0), #32
                                    05   12 0002F         BNEQ    2$
                               10   AE   D7 00031         DECL    DESC                                     ; 1582
                                    EA   11 00034         BRB     1$                                       ; 1579
                   6E               03   D0 00036 2$:     MOVL    #3, MSGVEC                               ; 1584
              04   AE 00000000G     8F   D0 00039         MOVL    #INPSMB$_JOBCARD, MSGVEC+4               ; 1585
              08   AE               01   D0 00041         MOVL    #1, MSGVEC+8                             ; 1586
              0C   AE          10   AE   9E 00045         MOVAB   DESC, MSGVEC+12                          ; 1587
                                    7E   7C 0004A         CLRQ    -(SP)                                    ; 1588
                                    64   DD 0004C         PUSHL   PUTMSG_ACTION_ROUTINE
                               0C   AE   9F 0004E         PUSHAB  MSGVEC
                   63               04   FB 00051         CALLS   #4, SYS$PUTMSG
                   52          04   AC   D0 00054 3$:     MOVL    SIG, R2                                  ; 1595
                   62               02   C2 00058         SUBL2   #2, (R2)
                                    7E   7C 0005B         CLRQ    -(SP)                                    ; 1596
                                    64   DD 0005D         PUSHL   PUTMSG_ACTION_ROUTINE
                                    52   DD 0005F         PUSHL   R2
                   63               04   FB 00061         CALLS   #4, SYS$PUTMSG
 04       04   A2  03               00   ED 00064         CMPZV   #0, #3, 4(R2), #4                        ; 1601
                                    10   12 0006A         BNEQ    4$
     7E       04   A2 10000000      8F   C9 0006C         BISL3   #268435456, 4(R2), -(SP)                 ; 1603
         00000000G 00               01   FB 00075         CALLS   #1, SYS$EXIT
                   50               01   D0 0007C 4$:     MOVL    #1, R0                                   ; 1607
                                         04 0007F         RET
```

; Routine Size: 128 bytes,   Routine Base: CODE + 0BF6

```
;  1511          1608  1 $INIT_STATE(DOLLAR_STATES, DOLLAR_KEYS);
;  1512        P 1609  1 $STATE(
;  1513          1610  1     ('$', TPA$_EXIT));
;  1514          1611  1
;  1515          1612  1
;  1516          1613  1 $INIT_STATE(JOB_STATES, JOB_KEYS);
;  1517        P 1614  1 $STATE(
;  1518          1615  1     ('JOB', TPA$_EXIT));
;  1519          1616  1
;  1520          1617  1
;  1521          1618  1 $INIT_STATE(EOJ_STATES, EOJ_KEYS);
;  1522        P 1619  1 $STATE(
;  1523          1620  1     ('EOJ', TPA$_EXIT));
;  1524          1621  1
;  1525          1622  1
;  1526          1623  1 $INIT_STATE(PASSWORD_STATES, PASSWORD_KEYS);
;  1527        P 1624  1 $STATE(
;  1528          1625  1     ('PASSWORD', TPA$_EXIT));
```

```
; 1530        1626  1 END
; 1531        1627  0 ELUDOM


                                                        .PSECT  _LIB$KEY1$,NOWRT,  SHR,  PIC,1

                                                00000 ;TPA$KEYSTO
                                                      U.5:      .BLKB   0
                                            42  4F  4A  00000 ;TPA$KEYST
                                                      U.7:      .ASCII  \JOB\                   ;
                                                FF  00003               .BYTE   -1              ;
                                                FF  00004 ;TPA$KEYFILL
                                                      U.10:     .BYTE   -1                      ;
                                                00005 ;TPA$KEYSTO
                                                      U.12:     .BLKB   0
                                            4A  4F  45  00005 ;TPA$KEYST
                                                      U.14:     .ASCII  \EOJ\                   ;
                                                FF  00008               .BYTE   -1              ;
                                                FF  00009 ;TPA$KEYFILL
                                                      U.17:     .BYTE   -1                      ;
                                                0000A ;TPA$KEYSTO
                                                      U.19:     .BLKB   0
                                44  52  4F  57  53  53  41  50  0000A ;TPA$KEYST
                                                      U.21:     .ASCII  \PASSWORD\              ;
                                                FF  00012               .BYTE   -1              ;
                                                FF  00013 ;TPA$KEYFILL
                                                      U.24:     .BYTE   -1                      ;

                                                        .PSECT  _LIB$STATE$,NOWRT,  SHR,  PIC,1

                                                00000 DOLLAR_STATES::
                                                              .BLKB   0
                                            1424  00000 ;TPA$TYPE
                                                      U.2:      .WORD   5156                    ;
                                            FFFF  00002 ;TPA$TARGET
                                                      U.3:      .WORD   -1                      ;
                                                00004 JOB_STATES::
                                                              .BLKB   0
                                            1500  00004 ;TPA$TYPE
                                                      U.8:      .WORD   5376                    ;
                                            FFFF  00006 ;TPA$TARGET
                                                      U.9:      .WORD   -1                      ;
                                                00008 EOJ_STATES::
                                                              .BLKB   0
                                            1500  00008 ;TPA$TYPE
                                                      U.15:     .WORD   5376                    ;
                                            FFFF  0000A ;TPA$TARGET
                                                      U.16:     .WORD   -1                      ;
                                                0000C PASSWORD_STATES::
                                                              .BLKB   0
                                            1500  0000C ;TPA$TYPE
                                                      U.22:     .WORD   5376                    ;
                                            FFFF  0000E ;TPA$TARGET
                                                      U.23:     .WORD   -1                      ;

                                                        .PSECT  _LIB$KEY0$,NOWRT,  SHR,  PIC,1
```

```
                        00000 DOLLAR_KEYS::
                                .BLKB   0
                        00000 ;TPA$KEY0
                              U.1:      .BLKB   0
                        00000 JOB_KEYS::
                                .BLKB   0
                        00000 ;TPA$KEY0
                              U.4:      .BLKB   0
                  0000* 00000 ;TPA$KEY
                              U.6:      .WORD   <U.5-U.4>                       ;
                        00002           .BLKB   2
                        00004 EOJ_KEYS::
                                .BLKB   0
                        00004 ;TPA$KEY0
                              U.11:     .BLKB   0
                  0000* 00004 ;TPA$KEY
                              U.13:     .WORD   <U.12-U.11>                     ;
                        00006           .BLKB   2
                        00008 PASSWORD_KEYS::
                                .BLKB   0
                        00008 ;TPA$KEY0
                              U.18:     .BLKB   0
                  0000* 00008 ;TPA$KEY
                              U.20:     .WORD   <U.19-U.18>                     ;

                                        .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY
;
;       Name                    Bytes                   Attributes
;
;  DATA                         1408  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE                         3190  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  _LIB$KEY0$                     10  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  _LIB$STATE$                    16  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  _LIB$KEY1$                     20  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)


;                               Library Statistics
;
;                                       --------- Symbols ---------       Pages       Processing
;       File                            Total     Loaded    Percent       Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1         18619        214          1         1000       00:01.9
;  _$255$DUA28:[SYSLIB]TPAMAC.L32;1         42         19         45           14       00:00.2


; Information:  1
; Warnings:     0
; Errors:       0
```

```
;                                   COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:INPSMB/OBJ=OBJ$:INPSMB MSRC$:INPSMB/UPDATE=(ENH$:INPSMB)

; Size:           2736 code + 1908 data bytes
; Run Time:          00:59.0
; Elapsed Time:      02:02.2
; Lines/CPU Min:       1654
; Lexemes/CPU-Min:  33073
; Memory Used:  411 pages
; Compilation Complete
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY